İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

A COMPUTATIONAL STUDY OF A FORWARD-FACING NOSE CAVITY ON A BLUNT-NOSED PROJECTILE IN SUPERSONIC FLOW USING FINITE VOLUME METHOD

M. Sc. Thesis by

Taner BALKAN, System Eng.

Department : Defense Technologies

Programme: Energy

JULY 2002

İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

A COMPUTATIONAL STUDY OF A FORWARD-FACING NOSE CAVITY ON A BLUNT-NOSED PROJECTILE IN SUPERSONIC FLOW USING FINITE VOLUME METHOD

M. Sc. Thesis by

Taner BALKAN, System Eng.

(514001020)

Date of submission : 24 July 2002

Date of defence examination: 30 July 2002

Supervisor (Chairman): Assoc. Prof. Dr. F. Oğuz EDİS

Members of the Examining Committee Prof. Dr. Kadir KIRKKÖPRÜ

Asst. Prof. Dr. Bülent YÜCEİL

JULY 2002

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

UCUNDA OYUK BULUNAN KÜT BURUNLU BİR MERMİ ETRAFINDAKİ AKIŞIN SÜPERSONİK HIZLARDA SONLU HACİMLER YÖNTEMİ İLE İNCELENMESİ

YÜKSEK LİSANS TEZİ

Sistem. Müh. Taner BALKAN

(514001020)

Tezin Enstitüye Verildiği Tarih : 24 Temmuz 2002
Tezin Savunulduğu Tarih : 30 Temmuz 2002

Tez Danışmanı : **Doç. Dr. F. Oğuz EDİS**
Diğer Jüri Üyeleri **Prof. Dr. Kadir KIRKKÖPRÜ**
Yrd. Doç. Dr. Bülent YÜCEİL

TEMMUZ 2002

ACKNOWLEDGMENTS

First of all, I would like to express my deepest gratitude to the Army for its support at every step of my education.

Then, I would like to express my sincere thanks and gratitude to my supervisor Assoc. Prof. Dr. F.Oğuz EDİS for his guidance, support and creative suggestions during the course of this study. I also extend thanks to Prof. Rüstem ASLAN and Prof. Kadir KIRKKÖPRÜ for their guidance and cooperation in Defense Technologies Program. I would also like to thank Dr. Bülent YÜCEİL and İdil FENERCİOĞLU (M.Sc.) for their valuable contributions about the experimental procedure. Thanks to the staff of Aerospace Engineering Department for their valuable help, advice and friendship. I feel lucky I had the chance to study with them.

Taner BALKAN JULY, 2002

TABLE OF CONTENTS

LIST OF TABLES

Page No

LIST OF FIGURES

LIST OF SYMBOLS

a_s : Speed of sound
D : Cavity diameter
D_n : Nose diameter
E : Total Energy
f_p : Primary resonance frequency
h : Cell area
G : Amplitude
H : Total Entalphy
L : Cavity depth
M : Mach number
R : Specific gas constant
R_n : Nose radius
Re : Reynolds number
P_0 : Total (Stagnation) pressure
P : Static Pressure
$\overline{P}$: Average Static Presure
P_∞ : Freestream Pressure
Q_k : Flux velocity
Q : Spatial discretization operator
T_0 : Total (Stagnation) Temperature
T : StaticTemperature
u,v : Cartesian Velocity Componenets
γ : Ratio of specific heats (for air, $\gamma = 1.4$)
δ : Shock standoff distance
λ_p : Primary resonance wavelength
μ : Dynamic viscosity
ρ : Density
$\supset$: Diagonal matrix
$\cap t$: Time step
$\cap x$: Interval
$a \cap t / \cap x$: Courant number

A COMPUTATIONAL STUDY OF A FORWARD-FACING NOSE CAVITY ON A BLUNT-NOSED PROJECTILE IN SUPERSONIC FLOW USING FINITE VOLUME METHOD

SUMMARY

Armies are interested in the kinetic energy projectiles because of their penetration characteristics. The more kinetic energy delivered, the more damage done to the target. This places premium on achieving high speed, since kinetic energy depends on the square of weapon's velocity. Currently, speeds of 1.5-2 km/s at sea level can be attained without ablation of the projectile tip but above 2 km/s, extremely high heating rates can cause tip ablation.

Introducing an axial cavity in the nose region of a supersonic projectile result in a local reduction in peak heating. Strong pressure oscillations are generated within the cavity to induce bow shock oscillations ahead of the projectile, which provide cooling mechanism at sea level.

The first objective of this study is to obtain the steady-flowfield results for a forward-facing nose cavity on a blunt-nosed projectile both at Mach 5 and at Mach 2. The second objective is to obtain the time-accurate simulations of pressure oscillations for L/D = 0.23 configuration at Mach 5.The FLUENT solver, a finite volume code, was used in this computational study in order to achieve these two objectives. All of these computations were performed using High-Performance Computing Systems of ITU.

Static pressure contours obtained by the computational analysis were compared with the experimental flow visualization results qualitatively. The computational results showed very good agreement with the experimental flow visualization results in terms of shock positions and shapes.

Time-accurate simulations showed that freestream noise in a small bandwidth of frequencies near the primary mode is the mechanism that drives resonant pressure oscillations within shallow forward-facing cavities.

UCUNDA OYUK BULUNAN KÜT BURUNLU BİR MERMİ ETRAFINDAKİ AKIŞIN SÜPERSONİK HIZLARDA SONLU HACİMLER YÖNTEMİ İLE İNCELENMESİ

ÖZET

Ordular zırh delme özellikleri nedeniyle kinetik enerji mermileri ile ilgilenmektedirler. Daha yüksek kinetik enerji, hedefte daha büyük tahribatı gerçekleştirir. Kinetik enerji, mermi hızının karesiyle orantılı olduğu için hız özelliği önem kazanmaktadır.Günümüzde deniz seviyesinde burunda erime olmadan 1.5-2 km/s hıza ulaşılmıştır fakat, 2 km/s'den daha yukarı hızlarda yüksek sıcaklıktan dolayı burunda erime oluşmaktadır.

Süpersonik bir merminin burnunda eksenel bir oyuk açılması bölgesel bir sıcaklık azalmasına neden olur. Oyuğun içerisinde, merminin önündeki şok dalgasının salınımına yol açan güçlü basınç salınımları oluşur, bu da deniz seviyesinde soğutma mekanizmasına sebep olur.

Bu çalışmanın ilk amacı, ucunda oyuk bulunan küt burunlu bir merminin 5 Mach ve 2 Mach hızlarındaki daimi akış sonuçlarını elde etmektir.Diğer amacı ise, 5 Mach hızındaki L/D = 0.23 konfigürasyonu için zamana bağlı basınç salınımlarını elde etmektir.Bu çalışmada belirtilen amaçlara ulaşmak için sonlu hacimler yöntemine dayalı FLUENT yazılımı kullanılmıştır. Tüm hesaplamalar İTÜ'nün Yüksek Başarımlı Hesaplama Sistemleri kullanılarak gerçekleştirilmiştir.

Bu hesaplamalı analiz sonucunda elde edilen basınç dağılımları, deneysel çalışmaların sonucu elde edilen akım görüntüleri ile niteliksel olarak karşılaştırılmıştır. Hesaplamalı analiz sonuçları, deneysel çalışma sonucu elde edilen akım görüntüleri ile şok yeri ve şekli açısından çok uyumludur.

Zamana bağlı simülasyonlar sığ derinlikteki oyuklarda ana moda yakın küçük band genişliğindeki frekanslarda deney gürültüsünün, rezonant basınç salınımları oluşturduğunu göstermiştir.

1. INTRODUCTION

" In peace prepare for war, in war prepare for peace. The art of war is of vital importance to the state. It is a matter of life and death, a road either to safety or to ruin. Hence under no circumstances can it be neglected … " Sun TZU [1]

The Army wants leap-ahead results, not incremental improvements and it wants them soon. Because of the ambitious timeline and technical uncertainties, the Army's plan is to rapidly identify the most promising technologies and then invest significant resources into them in hopes of obtaining leap-ahead results to be prepared for war.

The development of hypervelocity anti-armor projectiles is a promising technology. That is the reason why U.S budget contains $88 million for hypervelocity missiles that are small and light and designed to give lightly armored forces the lethal power now enjoyed only by tank units. [2] Not only U.S but also Sweden plans to make an anti-tank missile, a hypersonic kinetic energy weapon, named Bofors' Buster. It is intended to fly at more than 2000 meters per second and penetrate frontal protections. The project was started in 1987. Phase 1 and 2 feasibility studies carried out between 1990 and 1992 were followed by a phase 3A study which looked at the possibility of creating a missile able to engage armored vehicles, aircraft and helicopters. Such a dual role proved impractical, so Phase 3B concentrated on an anti-tank system capable of being mounted on a CV90 (Main Infantry Combat Vehicle used in Sweden). The missile would be fired from a remotely controlled eight-round launcher mounted above the turret. After being boosted to hypersonic speed by a rocket booster, the slim tungsten carbide penetrator would separate and fly out to the target. During the boost phase and unpowered flight the missile would be under the control of a guidance beam, which would steer it to impact on targets at ranges from 400 to 4000 meters. If selected for development, the Buster would become Sweden's long-range anti-tank weapon from 2010 onwards. [3]

As J.D. Anderson expressed in his book of Fundamentals of Aerodynamics [4], high-speed, supersonic flight had become a dominant feature of aerodynamics by the end of World War II. By this time, aerodynamicists appreciated the advantages of using slender, pointed body shapes to reduce the drag of supersonic vehicles. The more pointed and slender the body, the weaker the shock wave attached to the nose and hence the smaller the wave drag. Consequently, German V-2 rocket used during the last stages of World War II had a pointed nose, and all short-range rocket vehicles flown during the next decade followed suit. Then in 1953, the first hydrogen bomb was exploded by the United States. This immediately spurred the development of long-range intercontinental ballistic missiles (ICBMs) to deliver such bombs. These vehicles were designed to fly outside the region of the earth's atmosphere for distances of 8000 km or more and to 6700 m/s. At such high velocities, the aerodynamic heating of the reentry vehicle becomes severe, and this heating problem dominated the minds of high-speed aerodynamicists.

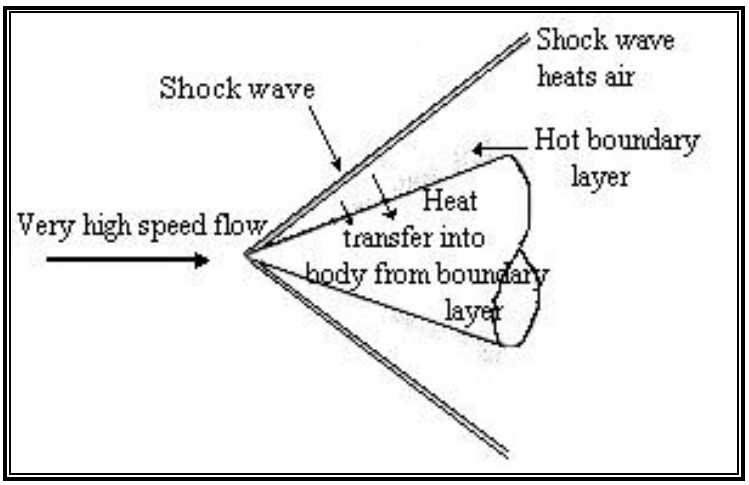


Fig 1.1: Energy of reentry goes into heating both the body and the air around the body. [4]

In 1951, H. Julian Allen at the NACA Ames Aeronautical Laboratory created one of those major breakthroughs that come very infrequently in engineering. He introduced the concept of the blunt reentry body. His thinking was paced by the following concepts. At the beginning of reentry, near the outer edge of the atmosphere, the vehicle has a large amount of kinetic energy due to its high velocity and a large amount of potential energy due to its high altitude. However, by the time the vehicle reaches the surface of the earth, its velocity is relatively small and its altitude is zero;

hence, it has virtually no kinetic or potential energy. Where has all the energy gone? The answer is that it has gone into heating the body and heating the airflow around the body. This is illustrated in Fig 1.1. Here, the shock wave from the nose of the vehicle heats the airflow around the vehicle; at the same time, the vehicle is heated by the intense frictional dissipation within the boundary layer on the surface. Allen reasoned that, if more of the total reentry energy could be dumped into the airflow, than less would be available to be transferred to the vehicle itself in the form of heating.

In turn, the way to increase the heating of the airflow is to create a stronger shock wave at the nose, i.e., to use a blunt-nosed body. The contrast between slender and blunt reentry bodies is illustrated in Fig. 1.4. This was a stunning conclusion to minimize aerodynamic heating; you actually want a blunt rather than a slender body. The result was so important that it was bottled up in a secret government document. [4]

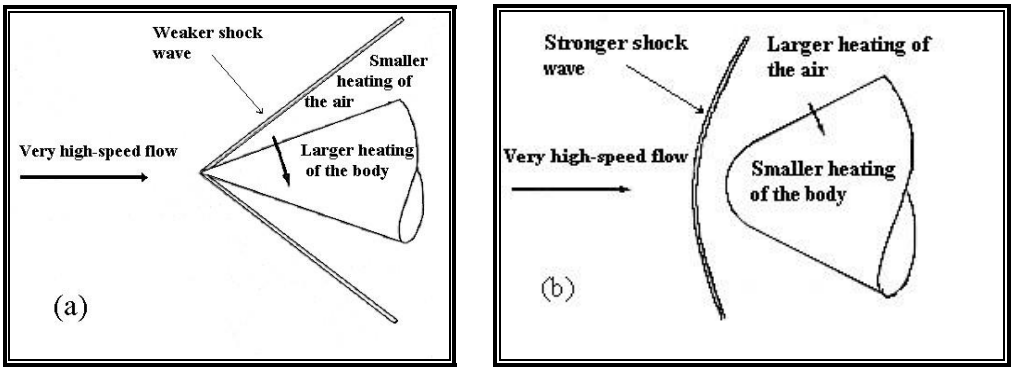


Fig 1.2: Contrast of aerodynamic heating for slender and blunt reentry vehicles. (a) Slender reentry body. (b) Blunt body. [4]

Increased penetration characteristic, which makes the development of hypervelocity anti-armor projectiles promising result largely from the extremely high kinetic energy at impact. Speeds between 1.5 km/s to 2 km/s at sea level are possible without melting or ablation of the projectile tip if high temperature-resistant materials such as tungsten are used, but at speeds above 2 km/s heating becomes a serious problem. As an example of the severity of the heating, the stagnation temperature at sea level and a velocity of about 2.6 km/s (Mach 7.6) corresponds to the melting of tungsten (3683 K). There is consequently a need to develop techniques to reduce nose tip heating rates if projectiles flying at speeds above 2 km/s are to be practical.

Guided missiles and projectiles generally have blunt nose for the reason that I've mentioned before. Previous studies [5] have been made on the subject to see whether it was possible to reduce the peak heating over the nose section of high speed projectiles. It has been shown that the nose heating is lower for cases with concave nose shapes than that for convex ones. Using this idea, there has been an experimental study at Mach 5 [6,7] in which a streamwise cavity is opened in front of the blunt nose to see whether it can alter the characteristics of the local boundary layer to reduce the nose heating. It was observed that the detached shock oscillates and for some cases the heat flux to the nose can be decreased. This fact can be used in the nose design by locating the forward-looking sensor at the base of the cavity and hence; the heat transfer and temperature rise of the window material may be reduced enough for some applications to eliminate the need for active cooling. Besides this study, another experimental study has been conducted in a supersonic blowdown wind tunnel at Mach 2 [8] .The main objective of this study was to determine if a single forward-facing axial cavity in front of a blunt nose would influence the aerodynamic forces acting on the body due to the oscillation of the detached shock wave. These two experiments that are carried out at Mach 5 and at Mach 2 are important to validate the present computational study. Conducting reliable experiments is a very important key in obtaining relevant information that will be used in CFD studies. Moreover, results from experiments can validate numerical work, which, in turn, may provide additional useful information, which cannot be obtained very easily from experiment.

The first objective of this study is to obtain the steady-flowfield results for a forward-facing nose cavity on a blunt-nosed projectile both at Mach 5 and at Mach 2. The second objective is to obtain the time-accurate simulations of pressure oscillations for cavity depth to diameter ratio $L/D = 0.23$ configuration.

To create the geometry and meshing the model GAMBIT pre-processor is used. FLUENT 5.5 is the solver that has been chosen for the computational study. Once a grid has been read into FLUENT, all remaining operations are performed within the solver. These include setting boundary conditions, defining fluid properties, executing the solution, refining the grid, and viewing and post processing the results.

FLUENT uses a control-volume-based technique to convert the governing equations to algebraic equations that can be solved numerically.

The FLUENT solver also allows for parallel processing and provides tools for checking and modifying the parallel configuration. Since the present work involved analysis of a compressible and complex flow that needs huge amount of memory and speed, it was a necessity to solve this problem using parallel processors. At this stage, 8-processor Artemis and 4-processor Blue, the High-performance Computing Systems in ITU, is used to perform the calculations.

The work done is presented in 6 chapters. Chapter 2 presents a brief review of the previous studies, which are related to the current problem. General description of the finite volume method and numerical scheme that is used in the coupled explicit solver of the FLUENT software are given in Chapter 3. Problem description, numerical assumptions and numerical procedure are described in Chapter 4. We can briefly name this chapter as computational representation of the problem. The results are presented and discussed in comparison with the experimental results in Chapter 5. The conclusions of the study are drawn in Chapter 6.

In the Appendix A, detailed information is given about creating the geometry and mesh generation. Parallel processing in FLUENT Software is explained in Appendix B. Finally; User Defined Functions that can be used to enhance the standard features of FLUENT are presented in Appendix C.

2. LITERATURE REVIEW

In 1959, Stallings and Burbank [5] argued that the stagnation heat transfer is a function of the local velocity gradient and can be reduced by increasing the degree of bluntness. Following this argument, various degrees of nose blunting have been used to alleviate the high heat transfer rates in the stagnation region of hypersonic missiles. After having finished their experiments in the NASA Langley Research Center, they reported an important fact that the stagnation point heat transfer rate for a concave-nose cylinder at supersonic Mach Numbers is considerably lower than that of a convex nose cone.

More recent studies [9-12] have focused on a specific nose cavity configuration such as missile/seeker configurations in order to make use of outcome of reduced heat transfer. If the reduction in heating was large enough then the forward-looking sensor could be placed at the base of the cavity without any active cooling, reducing the expense and complexity. This charming aim was the main reason for the mentioned studies. The schematic view of missile/seeker configuration was drawn consulting reference [12]

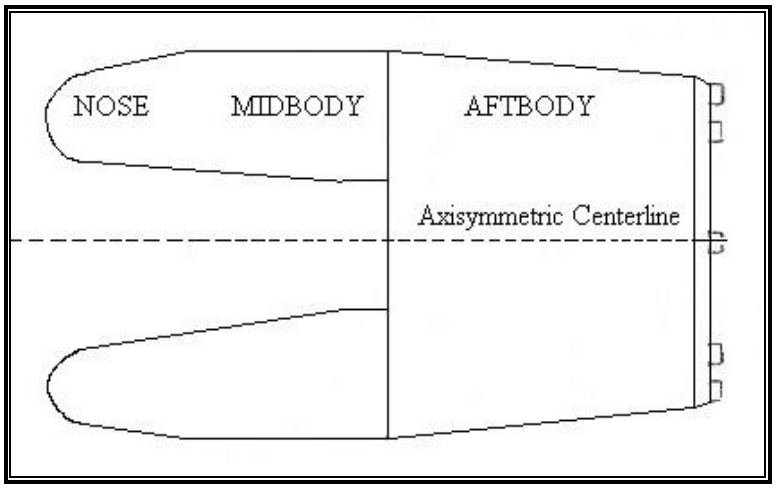


Fig. 2.1: The schematic view of missile/seeker configuration

Huebner and Utreja [9] reported that shock oscillation frequency and amplitude are directly related to cavity depth. Pressure uniformity on the cavity sidewall exists in all runs and significant reductions in heat flux are seen at the cavity base as compared to the nose rim.

Sambamurthi et al. [10] predicted a shock that oscillated at a frequency corresponding to the acoustic frequency associated with the cavity depth analyzed such that the wavelength was four times the cavity-base to shock distance. It was both numerical and experimental study in which the computational fluid dynamics code RAVENS, (Reacting Analysis of viscous Equations Navier-Stokes), that solves the time dependent, three-dimensional Navier-Stokes equations with non-equilibrium chemistry, and turbulence is used.

Marquart et al. [11] focused on the dynamics of the detached bow shock and the acoustic resonance in the forward facing nose cavity of a blunt faced model. They found that the primary mode of pressure oscillation in the cavity is at the classical organ pipe frequency, and the root mean square (rms) levels of the fluctuating pressure along the cavity wall increased toward the cavity base. They also found that the pressure variations drove the bow shock oscillations.

Huebner and Utreja [12] discussed the bow-shock behavior on the nose-cavity configuration represented in Fig. 2.1 at Mach 10. They showed that the fundamental acoustic frequency of the cavity is inversely proportional to the distance from the cavity base to the mean shock position and the mean shock-standoff distance was a function of the nose size. Since the mean shock standoff distance was not a function of the cavity depth, they suggested that the flow sees the nose-cavity configuration as if it possesses a flat face. They further showed that varying the angle of attack did not significantly alter the shock shape, standoff distance and the amplitude.

Bastianon [13] found that at Mach 3 for cavity depths greater than 40% of the cylindrical body radius, the viscous solution is unsteady and the flow generates an undamped, cyclically repeating oscillating wave.

Bohachevsky and Kostoff [14] reported that the flow about a conical cavity at Mach 5 has initial shock oscillations that are strongly damped and eventually reach a nonoscillatory condition. They further investigated a thin-walled hollow cylinder at Mach 10 incorporating artificial viscosity and found that the shock once again exhibited damped oscillations before eventually reaching a steady-state position. They also noted that the shock oscillation frequency was related to the distance from the cavity base to the shock equilibrium position.

Another numerical study was conducted by Yang and Antonison to investigate the oscillatory bow-shock behavior observed experimentally in front of concave nose cavities in hypersonic flow [15]. A time-accurate, multi-dimensional, density-based Navier-Stokes code, Fastran, was used with an inviscid assumption. They detected a periodic pressure oscillation with almost constant amplitude at the center of the cavity that implies the oscillatory motion of the bow shock. The numerical calculations also indicated that the oscillations were very sensitive to the cavity geometry. They found that the strength of oscillatory bow shock motion decreases as the cavity depth decreases.

Engblom et al. [16] carried out a numerical and an experimental study simultaneously. They studied the forward facing cavity flows. Approximately the same body configurations and freestream conditions were used in both the experiments and numerical simulations to provide a direct comparison of results. The flow is assumed to be axisymmetric. Inca, a commercial code, was selected for its ability to predict peak heating for the baseline case. Inca is a finite volume code that utilizes flux splitting with upwinding to capture strong shocks. They used the numerical input noise from experimental data to study the mechanism of resonance. They reported that resonant pressure oscillations occurred numerically only if the freestream fluctuations were present and that the oscillation strength increased with cavity depth. They found a good agreement between experiments and computations for the flowfield structure and surface heating. Sharp lips produce a recirculation region that cools the outer surface and severe heating just inside the cavity. Rounding the lip eliminates the recirculation region and alleviates heating inside the cavity. Numerical study results indicated that the flowfield structure and outer surface heating were insensitive to cavity depth. Experimental results hint that the strong

oscillations with deep cavities may produce an additional cooling effect. Numerical results also showed that the recirculation zone consists of one primary vortex and one small secondary vortex near the lip. Experimental and numerical results indicated that the flowfield structure and surface heating rates are sensitive to the lip radius. Numerical study showed that as the lip radius increases, the recirculation becomes smaller and the secondary vortex disappears.

Surface temperatures on a hemi-sphere cylinder body with a nose cavity in Mach 4.9 airflow have been measured using an infrared (IR) camera by Yuceil and Dolling [6]. Fluctuating surface pressures have also been measured at the cavity base. The cavity diameter *D* was fixed at one-half the cylinder diameter and the length *L* of the cavity was varied. If the cavity lip is sharp and the cavity is shallow (0.15P*L/D*P0.35) or very deep (*L/D*Σ1) an axisymmetric, nominally steady cool ring forms on the external surface downstream of the lip. Flow visualization shows that the cool ring is caused by separation at the lip. Rounding the cavity lip eliminates or reduces separation and temperatures return to levels characteristic of the model without the cavity. For "intermediate deep cavities (0.40 P *L/D*P0.70) the cavity pressure signals switch from a low-amplitude to high-amplitude level at random intervals resulting in an unstable, nonaxisymmetric temperature field downstream of the lip. Changes in cavity base shape from spherical to flat have little effect on the temperature history for shallow and very deep cavities, whereas for intermediate depth cavities the effects are more significant.

In 1997, Yuceil and Dolling [7] made experiments to determine if the unsteady flow induced by a streamwise cavity in the nose of a blunt body could reduce mean surface heat transfer rates compared to the same body without a cavity. Measurements were made at Mach 5 and included surface temperatures obtained using an infrared camera, fluctuating pressures at the cavity base, and bow shock visualization using planar laser Mie scattering. Cavities with a length-to-depth ratio of about 2 appear very promising in terms of reduced heating. Cavities for which the length-to-depth ratio vary from about 0.4 to 0.7 exhibit unstable, nonaxisymmetric surface temperature and cavity pressure histories. Instantaneous shock visualization reveals nonaxisymmetric shocks that correlate with the unstable pressure histories.

Cavity resonance frequency decreases as the cavity depth increases and pressure fluctuations increase in amplitude. In all cases, the primary oscillation frequency agrees well with simple organ pipe theory.

Fenercioğlu [8] studied to determine if a single forward-facing axial cavity in front of a blunt nose would influence the aerodynamic forces acting on the body due to the oscillation of the detached shock wave. Experimental studies have been conducted to obtain this objective in a supersonic blowdown wind tunnel at Mach 2. Force measurement data were obtained by using time-series data acquisition from a 3-component strain-gage and Schlieren method was used for flow visualization. The experiments were done on five models with various cavity depths. Frequencies of the Pressure oscillations are in the range of 2-7 kHz for the *L/D* cases considered in this study. However, it is found that the balance system of 150 x 150 Trisonic Wind Tunnel is not responding fast enough to be sensitive to such high frequencies. The force measurements with the strain gage showed that opening a cavity does not have a significant effect on the mean values of the aerodynamic forces on the blunt body for the *L/D* cases considered in this study. The flow visualization results showed that when the cavity depth was *L/D* = 0.5, a highly unstable shock with a bulge occurs for all runs. For all the other cavity depths, a stable shock with constant detachment distance was observed.

3. THEORETICAL BACKGROUND

In solving fluid flow problems we need to be aware that the underlying physics is complex and the results generated by a CFD code are at best as the physics (and chemistry) embedded in it and at worst as good as its operator. Prior to setting up and running a CFD simulation, an operator should be aware of the fact that there is a stage of identification and formulation of the flow problem in terms of the physical and chemical phenomena that need to be considered. FLUENT uses a control-volume-based technique to convert the governing equations to algebraic equations that can be solved numerically. In the light of these realities, general description of the Finite Volume Method is made in the beginning of this chapter. Then Jameson-Schmidt-Turkel Algorithm that is used in the Coupled-Explicit solver of the FLUENT is discussed.

3.1 General Description of the Finite Volume Method

The conservation laws of fluid motion may be expressed mathematically in either differential form or integral form. When a numerical scheme is applied to the differential equation, the domain of solution is divided into discrete points, upon which the finite difference equations are solved. On the other hand, when the integral form of the equations is utilized, the domain of solution is divided into small volumes (or areas for a two-dimensional case). Subsequently, the conservation laws in integral form are applied to these elementary volumes. The integral methods include finite volume and finite element methods.

Before proceeding to the details of the finite volume schemes, it is important to state the differences between the differential and integral methods so that the advantages and disadvantages of each method can be identified. The discussion will be limited to two dimensions, although the conclusions are valid for three dimensions as well.

The finite difference equations that approximate the partial differential equations are solved within a rectangular domain at equally distanced discrete points. Since the majority of physical domains are irregular in shape, a coordinate transformation from a physical space to a computational space is performed where the computational domain is rectangular. However, even with the coordinate transformation available, domains, which are highly irregular, would create serious difficulties in accuracy and convergence of the solution. The reason is that the metrics and Jacobian of transformation and the corresponding gradients, which are used in the governing equations, may include numerical discontinuities if the grid system is not relatively smooth.

At this point, it may be concluded that, in general, the finite difference methods possess inherited weakness for highly complicated domains. On the other hand, finite volume (or finite element) schemes do not encounter such weakness. That is because the independent variables are integrated directly on the physical domain and, therefore, grid smoothness is no longer an important issue. Thus, the governing equations can be solved if only the domain can be successfully discretized into elements. The geometrical difficulty is now the concern of the grid generation routine and not of the finite volume solver. Furthermore, finite volume schemes do not require a structured grid, as is required of the finite difference schemes; therefore, for most applications, unstructured grids are used. It is also important to emphasize that, since the integral equations are applied directly on the physical domain, a coordinate transformation is no longer required. It is then clear that the finite volume methods have advantages over the finite difference method if the geometry of the domain is complicated. That is, finite volume schemes provide great flexibility, in that wide range of choices is available for the selection of discrete volumes. However, it should be noted that if the domain can be discretized into a smooth structured grid, the finite difference method would be a better choice due to its efficiency over that of the finite volume or finite element methods. [17]

3.2 Jameson-Schmidt-Turkel Algorithm

Jameson-Schmidt-Algorithm, a combination of a finite volume discretization in conjunction with carefully designed dissipative terms of third order and a Runge Kutta time stepping scheme, is used to yield an effective method for solving the Euler equations in arbitrary geometric domains in FLUENT Coupled Explicit solver. "Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes" paper that was presented in AIAA 14th Fluid and Plasma Dynamics Conference in 1981 [20] is partly duplicated, for the sake of convenience.

3.2.1 Finite Volume Scheme [20]

Let p, ρ, u, v, E and H denote the pressure, density, Cartesian velocity components, total energy and total enthalpy. For a perfect gas

$$E = \frac{p}{(\gamma - 1)\rho} + \frac{1}{2}\left(u^2 + v^2\right) \quad , \qquad H = E + \frac{p}{\rho} \tag{3.1}$$

where γ is the ratio of specific heats. The Euler equations for two-dimensional inviscid flow can be written in integral form for a region Ω with boundary $\partial\Omega$ as

$$\frac{\partial}{\partial t}\iint_{\Omega} w \quad dx \quad dy + \oint_{\partial\Omega} (f dy - g dx) = 0 \tag{3.2}$$

where x and y are Cartesian coordinates and

$$w = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix} \quad , \qquad f = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uH \end{pmatrix} \quad , \qquad g = \begin{pmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ \rho vH \end{pmatrix} \tag{3.3}$$

The discretization procedure follows the method of lines in decoupling the approximation of the spatial and temporal terms. The computational domain is

divided into quadrilateral cells as in Fig. 3.1, and a system of ordinary differential equations is obtained by applying equation (3.2) to each cell separately. The resulting equations can then be solved by several alternative time stepping schemes.

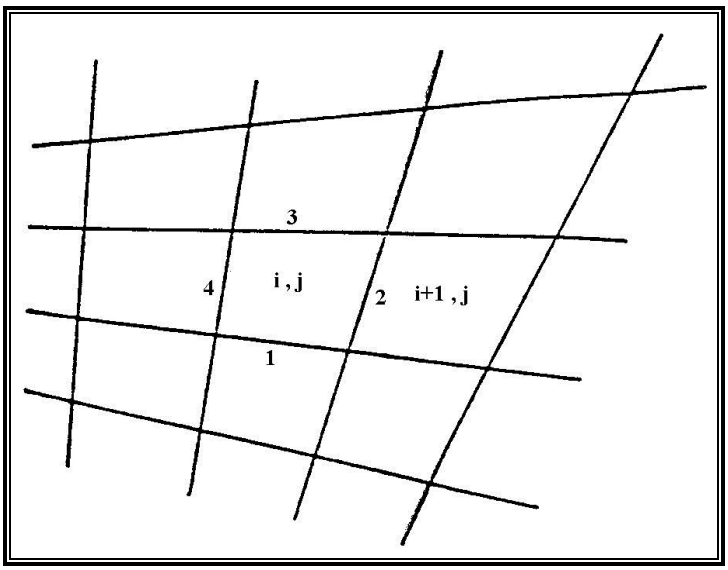


Fig. 3.1:The Computational domain that is divided into quadrilateral cells [20]

Let the values of the quantities associated with each cell be denoted by *i, j*. (These can be regarded as values at the cell center or average values for the cell). For each cell equation (3.2) assumes the form

$$\frac{d}{dt}(hw) + Qw = 0 \tag{3.4}$$

where h is the cell area, and the operator Q represents an approximation to the boundary integral defined by the second term of equation (3.2). This is defined as follows. Let $\cap x_k$ and $\cap y_k$ be the increments of x and y along side k of the cell, with appropriate signs. Then the flux balance for, say, the x momentum component, is represented as

$$\frac{\partial}{\partial t}(h\rho) + \sum_{k=1}^{4} \left(Q_k \ \rho u_k + \Delta y_k \ \ p_k \right) = 0 \tag{3.5}$$

where h is the cell area, Q_k is the flux velocity

$$Q_k = \Delta y_k \ \ u_k - \Delta x_k \ \ v_k \tag{3.6}$$

and the sum is over the four sides of the cell. Each quantity such as u_1 or $(\rho u)_1$ is evaluated as the average of the values in the cells on the two sides of the face,

$$(\rho u)_1 = \frac{1}{2}(\rho u)_{i,j} + \frac{1}{2}(\rho u)_{i+1,j} \tag{3.7}$$

for example. The scheme reduces to a central difference scheme on a Cartesian grid and is second order accurate provided that the grid is smooth enough.

3.2.2. Dissipative Terms [20]

To suppress the tendency for odd and even point decoupling, and to prevent the appearance of wiggles in regions containing severe pressure gradients in the neighborhood of shock waves or stagnation points, it proves necessary to augment the finite volume scheme by the addition of artificial dissipative terms. Therefore equation (3.4) is replaced by the equation

$$\frac{d}{dt}(hw) + Qw - Dw = 0 \tag{3.8}$$

where Q is the spatial discretization operator defined by equations (3.5-3.7), and D is a dissipative operator. Extensive numerical experiments have established that an effective form for D_w is a blend of second and fourth differences with coefficients which depend on the local pressure gradient.

The construction of the dissipative terms for each of the four dependent variables is similar. For the density equation

$$D\rho = D_x\rho + D_y\rho \tag{3.9}$$

where $D_x\rho$ and $D_y\rho$ are corresponding contributions for the two coordinate directions, written in conservation form

$$D_x\rho = d_{i+\frac{1}{2},j} - d_{i-\frac{1}{2},j} \quad ; \quad D_y\rho = d_{i,j+\frac{1}{2}} - d_{i,j-\frac{1}{2}} \tag{3.10}$$

The terms on the right all have a similar form, for example,

$$d_{i+\frac{1}{2},j} = \frac{h_{i+\frac{1}{2},j}}{\Delta t}\left\{\varepsilon^{(2)}_{i+\frac{1}{2},j}\,(\rho_{i+1,j}-\rho_{i,j}) - \varepsilon^{(4)}_{i+\frac{1}{2},j}\,(\rho_{i+2,j}-3\rho_{i+1,j}+3\rho_{i,j}-\rho_{i-1,j})\right\} \tag{3.11}$$

where h is the cell volume, and the coefficients ε^2 and ε^4 are adapted to the flow. Define

$$v_{i,j} = \frac{\left|p_{i+1,j} - 2\,p_{i,j} + p_{i-1,j}\right|}{\left|p_{i+1,j}\right| + 2\left|p_{i,j}\right| + \left|p_{i-1,j}\right|} \tag{3.12}$$

Then

$$\varepsilon^{(2)}_{i+\frac{1}{2},j} = k^{(2)} \max\left(v_{i+1,j}, v_{i,j}\right) \tag{3.13}$$

and

$$\varepsilon^{(4)}_{i+\frac{1}{2},j} = \max\left(0, \left(\kappa^{(4)} - \varepsilon^{(2)}_{i+\frac{1}{2},j}\right)\right) \tag{3.14}$$

where typical values of the constants $\nu^{(2)}$ and $\nu^{(4)}$ are $\nu^{(2)} = 1/4$, $\nu^{(4)} = 1/256$.

The dissipative terms for the remaining equations are obtained by substituting ⟨u, ⟨v and either ⟨E, ⟨x, ⟨H for ⟨ in these formulas.

The scaling $h / \cap t$ in equation (3.11) conforms to the inclusion of the cell area h in the dependent variables of equation (3.8). Since equation (3.11) contains undivided differences, it follows that if $\Pi^{(2)} = 0(\cap x^2)$ and $\Pi^{(4)} = 0(1)$, then the added terms are of order $\cap x^3$. This will be the case in a region where the flow is smooth. Near a shock wave $\Pi^{(2)} = 0(1)$, and the scheme behaves locally like a first order accurate scheme.

In smooth regions of the flow, the scheme is not sufficiently dissipative unless the fourth differences are included, with the result that calculations will generally not converge to a completely steady state. Instead, after they have reached an almost steady state, oscillations of very low amplitude continue indefinitely (with max |⟨ / |t Ξ10^3 , for example). These appear to be induced by reflections from the

boundaries of the computational domain. Near shock waves it has been found that the fourth differences tend to induce overshoots, and therefore they are switched off by subtracting $\Pi^{(2)}$ from $\nu^{(4)}$ in equation (3.14)

3.2.3. Time Stepping Schemes [20]

Stable time stepping methods for equation (3.8) can be patterned on standard schemes for ordinary differential equations. Multistage two level schemes of the Runge Kutta type have the advantage that they do not require any special starting procedure, in contrast to leap frog and Adams Bashforth methods, for example. The extra stages can be used either to improve accuracy or to extend the stability region. An advantage of this approach is that the properties of these schemes have been widely investigated and are readily available in textbooks on ordinary differential equations.

Consider a linear system of equations

$$\frac{dw}{dt} + Aw = 0$$

Suppose that A can be expressed as $A = T\supset T^{-1}$ where T is the matrix of the eigenvectors of A, and $\supset$ is diagonal. Then setting $v = v^{-1}w$ yields separate equations

$$\frac{d}{dt} v_k + \lambda_k \; v_k = 0$$

for each dependent variable v_k. The stability region is that region of the complex plane containing values of $|\cap t$ for which the scheme is stable. Consider now the model problem

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} + \varepsilon \Delta x \frac{\partial^2 u}{\partial x^2} = 0 \qquad (3.15)$$

on a uniform mesh with interval $\cap x$, with a dissipative term of order $\cap x$. This can be reduced to a system or ordinary differential equations by introducing central-difference approximations for $|/|x$ and $|^2/|x^2$:

$$\frac{du_i}{dt} + \frac{a}{\Delta x}(u_{i+1} - u_{i-1}) + \frac{\varepsilon}{\Delta x}(u_{i+1} - 2u_i - u_{i-1}) = 0$$

Taking the Fourier transform in space

$$\hat{u} = \frac{1}{2\pi}\int_{-\infty}^{\infty} ue^{iwx}\,dx$$

This becomes

$$\frac{d\hat{u}}{dt} + \lambda\hat{u} = 0$$

where

$$\lambda = \frac{1}{\Delta x}\left(i \quad a \quad \sin w\Delta x - 4\varepsilon \sin^2 \frac{w\Delta x}{2}\right)$$

It can be seen that the maximum allowable value of the imaginary part of $\Leftrightarrow\cap$t determines the maximum value of the Courant number $a\cap t/\cap x$ for which the calculation will be stable , while the addition of the dissipative term shifts the region of interest to the left of the imaginary axis.

In the present case, if the grid is held fixed in time so that the cell area h is constant, the system of equations (3.8) has the form

$$\frac{dw}{dt} + Pw = 0 \tag{3.16}$$

where if Q is the discretization operator defined in section 3.2.1 and D is the dissipative operator defined in section 3.2.2 , the nonlinear operator P is defined as

$$Pw \equiv \frac{1}{h}(Qw - Dw) \tag{3.17}$$

The investigation has concentrated on two time stepping shemes. The first is a three stage scheme which is defined as follows. Let a superscript *n* denote the time level and let $\cap t$ be the time step.

Then at time level n set

$w^{(0)} = w^{n}$

$w^{(1)} = w^{(0)} - \cap t \; Pw^{(0)}$

$w^{(2)} = w^{(0)} - \cap t/2 \; (Pw^{(0)} + Pw^{(1)})$

$w^{(3)} = w^{(0)} - \cap t/2 \; (Pw^{(0)} + Pw^{(2)})$

$w^{n-1} = w^{(3)}$

This scheme can be regarded as a Crank Nicolson scheme with a fixed point iteration to determine the solution at time level n+1 and the iterations terminated after the third iteration. It is second order accurate in time and for the model problem (3.15) with $\int = 0$, it is stable when the Courant number

$$\left|\frac{a\Delta t}{\Delta x}\right| \leq 2$$

This bound is not increased by additional iterations. Compared with standard third order Runge Kutta schemes, this scheme gives up third order accuracy in time for a larger bound on the Courant number.

The other scheme which has been extensively investigated is the classical fourth order Runge Kutta scheme, defined as follows. At time level n set

$w^{(0)} = w^{n}$

$w^{(1)} = w^{(0)} - \cap t/2 \; Pw^{(0)}$

$w^{(2)} = w^{(0)} - \cap t/2 \; Pw^{(1)}$

$w^{(3)} = w^{(0)} - \cap t \; Pw^{(2)}$

$w^{(4)} = w^{(0)} - \cap t/6 \; (P\mathrm{w}^{(0)} + 2\, P\mathrm{w}^{(1)} + 2\, P\mathrm{w}^{(2)} + P\mathrm{w}^{(3)})$

$w^{(n+1)} = w^{(4)}$

This scheme is fourth order accurate in time, and for the model problem (3.15) with $\int = 0$, it is stable for Courant numbers

$$\left| \frac{a \Delta t}{\Delta x} \right| \leq 2\sqrt{2}$$

Its stability region, also extends well to the left of the imaginary axis, allowing latitude in the introduction of dissipative terms. [21]

Both schemes have the property that if $P\mathrm{w}^{n} = 0$ then $w^{(1)} = w^{(0)}$, and so on , so that $w^{(n+1)} = w^{n}$ and the steady state solution is $P\mathrm{w} = 0$ independent of the time step $\cap t$. This allows a variable time step determined by the bound on the local Courant number to be used to accelerate convergence to a steady state without altering the steady state.

The expense of re-evaluating the dissipative terms at every stage of these schemes is substantial. One method of avoiding this is to introduce the dissipative terms in a separate fractional step after the last stage of the Runge Kutta scheme. Then equation (3.17) is replaced by

$$Pw \equiv \frac{1}{h} \; Qw \tag{3.17}$$

and the fourth order Runge Kutta scheme defined by equation (3.19) , for example, is modified by setting

$$w^{n+1} = w^{(4)} + \cap t \, D \; w^{(4)}$$

This method has the advantage that the stability properties for the two fractional steps are independent, so that the scheme will be stable if each fractional step is stable. It has the disadvantage that the steady state solution is no longer independent of the time step.

An alternative approach which has proved successful in practice, is to freeze the dissipative terms at their values in the first stage. Thus the fourth order Runge Kutta scheme is modified so that it has the form

$$w^{(0)} = w^{n}$$

$$w^{(1)} = w^{(0)} - \cap t/(2h)\, Q\mathrm{w}^{(0)} + \cap t/(2h)\, D\mathrm{w}^{(0)}$$

$$w^{(2)} = w^{(0)} - \cap t/(2h)\, Q\mathrm{w}^{(1)} + \cap t/(2h)\, D\mathrm{w}^{(0)}$$

$$w^{(3)} = w^{(0)} - \cap t/\mathrm{h}\ Q\mathrm{w}^{(2)} + \cap t/h\, D\mathrm{w}^{(0)}$$

$$w^{(4)} = w^{(0)} - \cap t/(6h)\ (\, Q\mathrm{w}^{(0)} + 2\, Q\mathrm{w}^{(1)} + 2\, Q\mathrm{w}^{(2)} + Q\mathrm{w}^{(3)}\,) + \cap t/h\ D\mathrm{w}$$

The operators Q and D require roughly equal amounts of computation. Assigning to each 1 unit of work and assuming that dissipative terms would be required in the leap frog or MacCormack schemes, both of which have maximum time steps bounded by a Courant number of one, one obtains the following table for the relative efficiency of the schemes.

Table 3.1: Relative efficiency of the schemes. [20]

Scheme	Evaluations of Q_w	Evaluations of D_w	Work	Max. Courant Number	Efficiency = time step/work
Leap frog	1	1	2	1	1/2
MacCormack	2	1	3	1	1/3
3 stage	3	3	6	2	1/3
4 stage	4	4	8	2,8	0,35
4 stage (frozen D_w	4	1	5	2,8	0,56

4. COMPUTATIONAL REPRESENTATION OF THE PROBLEM

The first objective of this study is to obtain the steady-flowfield results for a forward-facing nose cavity on a blunt-nosed projectile both at Mach 5 and at Mach 2. The second objective is to obtain the time-accurate simulations of pressure oscillations for L/D = 0.23 configuration at Mach 5. This chapter describes the problem and the computational arrangements made to achieve these two objectives.

4.1 Problem Description

4.1.1. Cavity Configurations

a) L/D = 0.23 Configuration.

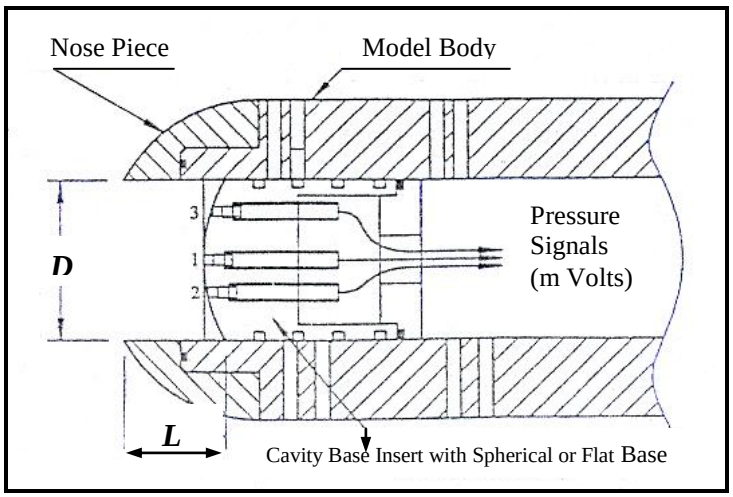


Fig 4.1: Schematic view of the model (L/D = 0.23). [6]

Yüceil and Dolling conducted the experiments in the Mach 5 blowdown wind tunnel of the University of Texas at Austin. 2.275 Mpa stagnation pressure and 370-375 K temperature are flow conditions.

The model is a hemispherically blunted cylinder, 5.08 cm in diameter, with a variable-length cavity at the tip (Fig 4.1). Since L/D = 0.23 configuration is studied in this computational study, L is considered as 0.5842 cm. Cavities are designated shallow, medium and deep: the term shallow is used if L/D is less than 0.40, medium if L/D is between 0.40 and 0.70, and deep if L/D is greater than 0.7. [6]

b) L/D = 0.5 Configuration.

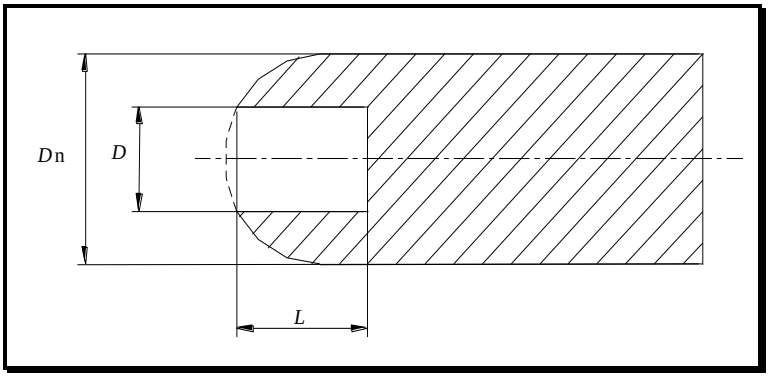


Fig 4.2: Schematic view of the model (L/D = 0.5). [8]

The second model was used to conduct experiments at the 150 x 150 Trisonic Wind Tunnel, located at the Istanbul Technical University Trisonic Research Center (TAM) by Fenercioğlu [8]. 2.5 x 10^5 Pa stagnation pressure and 289 K temperature are flow conditions.

The model geometry is a hemisphericallly blunted-cylinder with an axisymmetric streamwise circular nose cavity as shown schematically in Figure 4.2. D_n is the diameter of the blunt nose of the model body and D is the cavity diameter. These diameters are kept the same for all models: D_n = 3 cm and D = 1.5 cm. L, the cavity, is defined as the distance from the cavity lip to the cavity base-side wall junction. Since L/D = 0.5 configuration is studied in this computational study, L is considered as 0.75 cm.

Fenercioğlu [8] conducted experimental measurements to determine if a single forward-facing axial cavity in front of a blunt nose would influence the aerodynamic forces acting on the body due to the oscillation of the detached shock wave.

4.1.2. Finite Volume Code Description.

FLUENT, a commercial code for modeling fluid flow and heat transfer in complex geometries, was selected for this study on the basis of available documentation, technical support and the solution-adaptive grid capability that is particularly useful for accurately predicting flow fields in regions with large gradients. In comparison to solutions on structured or block-structured grids, this feature significantly reduces the time required to generate a "good" grid. Solution-adaptive refinement makes it easier to perform grid refinement studies and reduces the computational effort required to achieve a desired level of accuracy, since mesh refinement is limited to those regions where greater mesh resolution is needed.

Although FLUENT has different solution algorithms for flows, Coupled Explicit solver based on Jameson-Schmidt-Turkel (JST) algorithm was chosen considering the characteristics of the studied flow. The JST uses multi-stage Runge Kutta time stepping and switches locally to the first order scheme using artificial viscosity term to damp oscillations near discontinuities in the vicinity of shock waves. Since a supersonic flow problem has been solved in this study, the JST scheme proved very effective with its characteristic to damp oscillations in the vicinity of the shock waves.

4.2. Numerical Assumptions.

Several simplifying assumptions are made in the simulations. The models used in the experiments had either a spherical base shape or a flat base shape, but all numerical assumptions assumed a flat shape. This difference is not considered significant, since the behavior of the cavities in the experiments [6] demonstrated little sensivity to base shape, except possibility in the medium range of cavity *L/D*. In view of the small scale of the flowfield (i.e., the nose region), laminar flow is assumed. The freestream Reynolds number is roughly 5.0×10^7/m. The actual Reynolds number (per meter) is much smaller along the body surface inside the cavity and outside the cavity near the lip, because of the low-speed flow. The wall temperature is assumed isothermal (T_{wall} = 300 K) and the flow is assumed calorically perfect considering the previous numerical studies [16]. The models are axisymmetric. Experiments [6,8] showed that flow domain is axisymmetric in the steady-state condition as well.

4.3. Numerical Procedure.

4.3.1. Steady-flowfield Solution Procedure.

Flow domain and boundary conditions set in the study can be seen in Fig. 4.3. Pressure far-field conditions were used to model a free-stream condition at infinity, with free-stream Mach number and static conditions being specified. Static Temperature and Pressure values are needed to apply pressure far-field boundary conditions. Since we know the stagnation pressure and temperature from the experiments, we can calculate them from the isentropic flow equations:

$$\frac{P_0}{P} = \left(1 + \frac{\gamma - 1}{2} M^2\right)^{\gamma/\gamma - 1} \tag{4.1}$$

$$\frac{T_0}{T} = 1 + \frac{\gamma - 1}{2} M^2 \tag{4.2}$$

Equations (4.1) and (4.2) give the ratios of total (stagnation) to static pressure and temperature, respectively, at a point in the flow as a function of Mach number M at that point for γ =1.4 (which corresponds to air at standard conditions). Pressure far-field boundary conditions are given in Table 4.1.

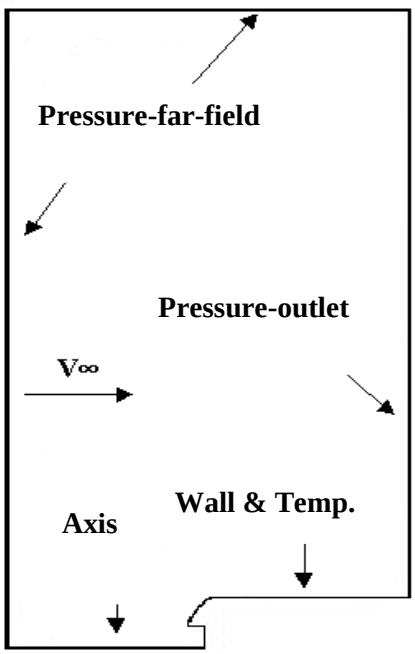


Fig 4.3: Flow domain and boundary conditions.

Table 4.1: Pressure far-field boundary conditions

	L/D = 0.23	L/D = 0.5
Stagnation Pressure (P_0)	227.5 kPa	250 kPa
Static Pressure (P)	4.3 kPa	31.95 kPa
Stagnation Temperature (P_0)	370 K	289.15 K
Static Temperature (T)	62 K	160.5 K
Mach Number	5	2

Wall boundary conditions were used to bound fluid and solid regions. Since we are solving the energy equation, we need to define thermal boundary conditions at wall boundaries. The wall temperature is assumed isothermal (T_{wall} = 300 K) previously. So the fixed temperature condition was selected in the Wall panel. Pressure outlet boundary condition is used to define the static pressure at flow outlet (and also other scalar variables, in case of backflow). Pressure outlet boundary conditions require the specification of a static (gauge) pressure at the outlet boundary. The value of static pressure specified is used only while the flow is subsonic. Should the flow become locally supersonic, the specified pressure is no longer used; pressure will be extrapolated from the flow in the interior. All other flow quantities are extrapolated from the interior. The axis boundary type was used as the centerline of the axisymmetric geometry.

In this computational study the solution-adaptive mesh refinement feature was used and because of this, unstructured triangular mesh was generated.

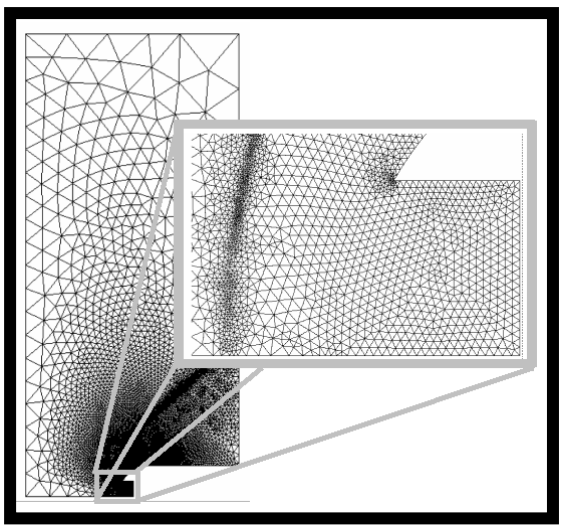


Fig 4.4: Unstructured triangular mesh that is used in the computational study.

Since the position of the shockwave was unknown at the beginning, fine mesh in the vicinity of the body and coarse mesh away from the body is generated initially. Then, mesh was refined step by step as the solution proceeded. Adapting on gradients of static pressure was chosen as the criterion for refining in the region of shock waves. Conformal mode that refines the cells by splitting the longest edge of the triangle was chosen as well. Smoothing and face swapping were done to complement grid adaption, usually increasing the quality of the final numerical mesh. Smoothing repositions the nodes and face swapping modifies the cell connectivity to achieve these improvements in quality.

In the procedure, periodic oscillations have been observed instead of convergence in some time intervals. Mesh-refinement was imposed in these conditions to observe the convergence more quickly.

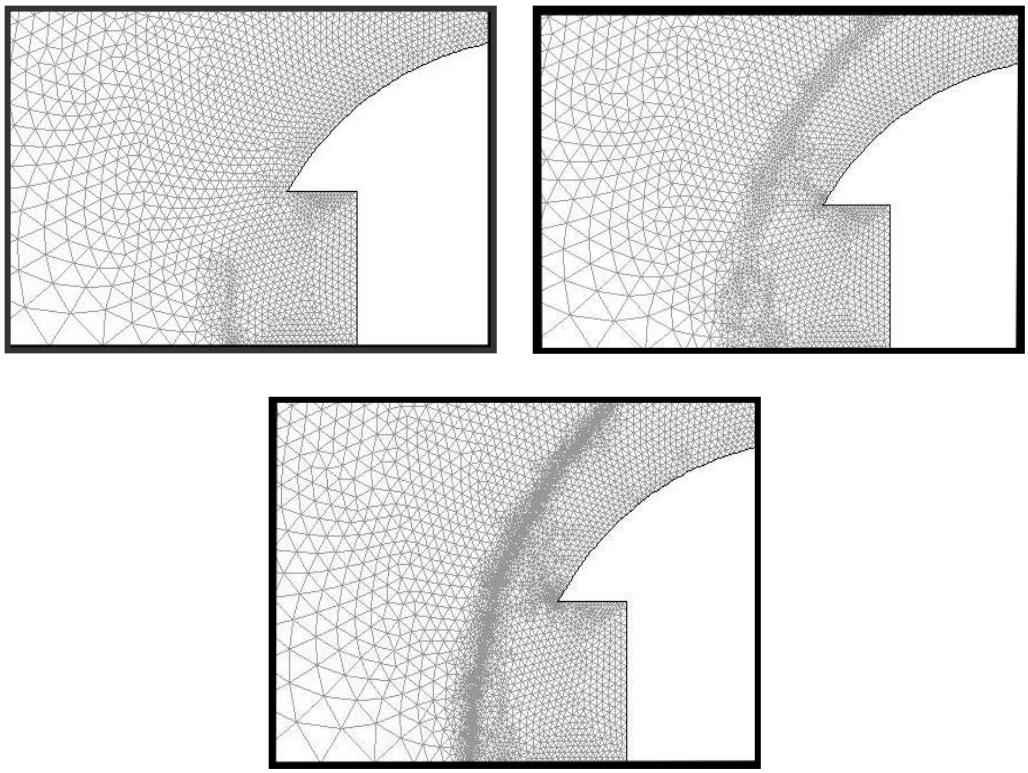


Fig 4.5: Solution-adaptive mesh refinement for *L/D* = 0.23 configuration.

3 different mesh examples from different stages of the solution procedure for *L/D* = 0.23 is shown in Fig 4.5.Apart from these, you can find information about the meshes and iteration numbers in Table 4.2. *L/D* = 0.23 problem has been solved with

Parallel Computation in Artemis using 3 processors. The solution process took approximately 26 hours in wall clock time. *L/D* = 0.5 problem has been solved in Blue using 4 processors and it took approximately 3 hours and 10 minutes to reach the converged solution. In both problems, the areas of minimum cell values in the final meshes decreased 10 times the initial minimum cell areas.

Table 4.2: Information about the meshes and iteration numbers.

L/D = 0.23, MACH 5			*L/D* = 0.5, MACH 2		
ITERATION #	MIN. CELL AREA (m^2)	CELL #	ITERATION #	MIN. CELL AREA (m^2)	CELL #
10000	2.065×10^{-8}	12691	3000	2.068×10^{-8}	12707
120000	1.982×10^{-8}	14050	4000	1.980×10^{-8}	13914
80000	1.948×10^{-8}	16305	14000	1.915×10^{-8}	16289
4000	1.933×10^{-8}	18651	2202	1.901×10^{-8}	18167
1760	1.748×10^{-8}	20062	2923	1.060×10^{-8}	20457
2344	1.701×10^{-9}	21752	3320	3.503×10^{-9}	21348

4.3.2. Time-accurate simulations of pressure oscillations.

Oscillating pressure levels within a cavity are a dominant experimental flow feature in forward-facing cavity configurations. It is well known from the previous studies [6,7,16] that cavity flows can resonate at certain frequencies depending on the geometry and the size of the cavity as a result of amplification of small disturbances present in the freestream. So it will be very helpful to understand the characteristics of these oscillations. The most important parameter for the dynamics of a streamwise cavity is the cavity depth. A cavity is said to be resonating if the frequencies of the measured pressure oscillations concentrate in narrow bands at specific values. Most of the energy of the oscillations is at the primary resonance frequency and it is usually the lowest band. Other resonance bands at higher frequencies are generally the harmonics of the primary resonance frequency. The primary resonance frequency can be calculated from a simple linear theory if the cavity depth and the speed of sound inside the cavity are known.

In the classical organ pipe model the wavelength λ_p of the primary resonance is given as,

$$\lambda_p = 4L \tag{4.3}$$

where L is the cavity depth. The frequency f_p corresponding to this wavelength can be calculated as,

$$f_p = \frac{a_s}{\lambda_p} \tag{4.4}$$

where a_s is the speed of sound inside the cavity.

Assuming that the gas temperature inside the cavity is approximately the stagnation temperature (T_0) of the flow, a_s can be calculated as,

$$a_s = \sqrt{\gamma R T_0} \tag{4.5}$$

where γ is the ratio of specific heats and R is the specific gas constant.

Combining equation (4.4) and equation (4.5),

$$f_p = \frac{\sqrt{\gamma R T_0}}{4L} \tag{4.6}$$

is obtained. Since the pressure waves travel between the shock wave and the cavity base, the appropriate L is the distance from the mean bow shock location to the cavity base. Therefore the shock standoff distance δ must be added to the cavity depth. It can be estimated using the correlations for hypersonic shock-wave shapes [19]. For a sphere-nosed body,

$$\frac{\delta}{R_n} = 0.143\, e^{\frac{3.24}{M_\infty^2}} \tag{4.7}$$

where R_n is the nose radius.

Figure 4.6 shows the predictions of Eq. (4.6). Note that Eq. (4.6) provides good agreement with values derived from experimental runs and time-accurate numerical simulations. [16] Previous time-accurate numerical simulations indicate that, for truly steady freestream conditions and the cavity depths (*L/D* P 1.25), the oscillations dissipate until a steady flow is achieved. However numerical resonance is obtained by introducing broad-bandwidth freestream noise (e.g., unsteady freestream pressure at the inflow boundary). The large fluctuations, high heat fluxes, and high optical distortion observed in conventional wind tunnels are spurious effects of the high freestream noise levels.

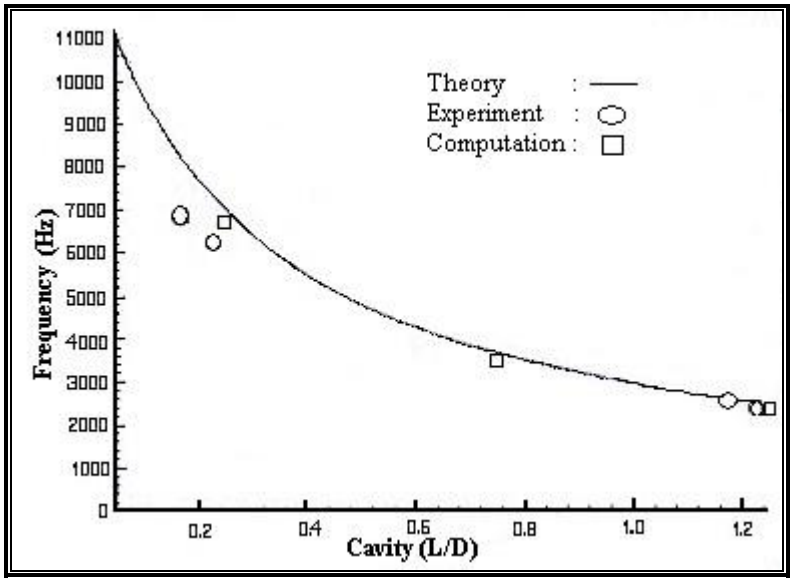


Fig.4.6: Primary mode frequency vs. cavity *L/D* (*D* = 2.54 cm) [16]

In this computational study, a sinusoidal freestream inflow pressure perturbation, with a peak to peak amplitude of H0.02 $P_э$ was simulated using FLUENT solver to understand the resonance mechanism for *L/D* = 0.23 shallow configuration. In order to do this a UDF was written in C code (it's detailed explanation can be found in Appendix C). The primary mode was estimated at 7000 Hz from Eq. (4.6). Beginning from this frequency, 6250, 5800 and 20000 Hz. frequencies were tried as well in order to determine the distinctive resonant frequency.

Time-accurate numerical simulations of pressure oscillations within the cavity were conducted using the steady-flowfield solution as an initial condition followed by

explicit time stepping. The Interpolate Data panel in FLUENT solver gave the opportunity to interpolate solution data from one grid to another. The mesh in the shockwave movement area was refined by using region and volume adaption options of the FLUENT solver. The area of minimum cell value of the final mesh is $1.034 \text{ X } 10^{-9}$. Figure 4.7 displays the final mesh used in time-accurate simulations of the pressure oscillations.

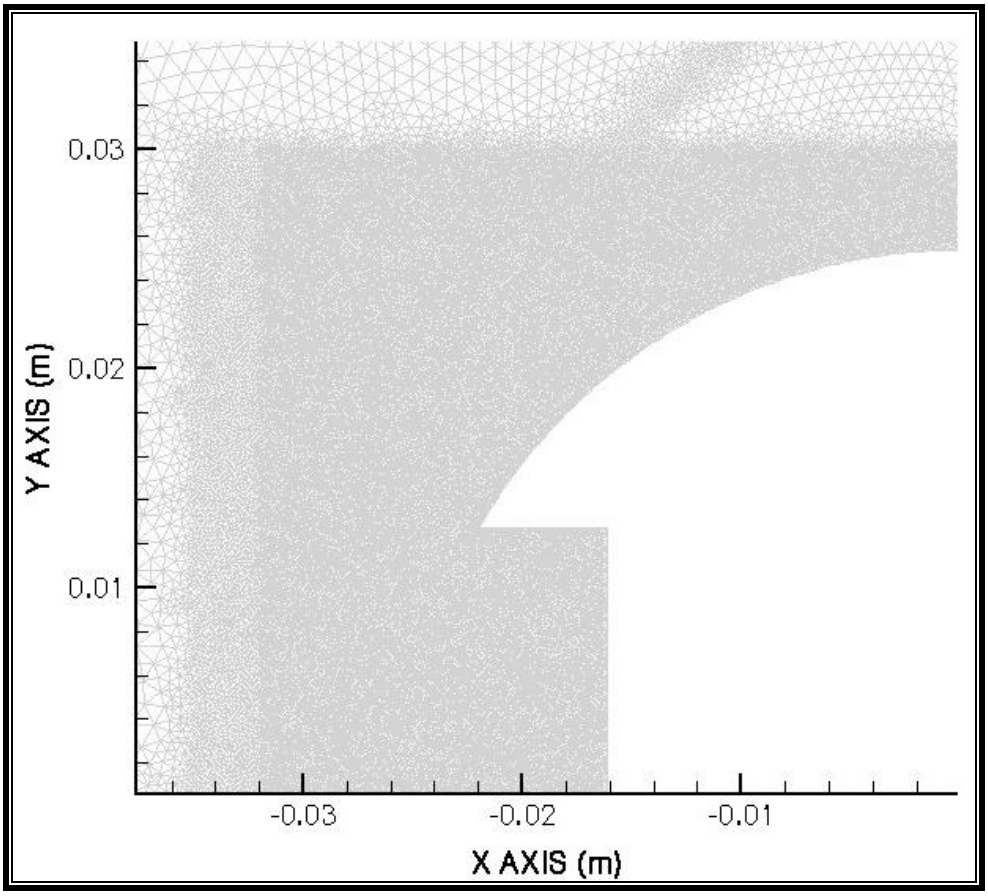


Fig.4.7: Final mesh used in time-accurate simulations of the pressure oscillations.

Two points were chosen in the pressure far field boundary region and on the base of the cavity to monitor the inflow pressure and cavity base pressure respectively.

5. RESULTS AND DISCUSSION

Two main types of computational studies were performed during the course of this study; steady-flowfield results for a forward-facing nose cavity on a blunt-nosed projectile (both at Mach 5 and at Mach 2) and the time-accurate simulations of pressure oscillations for *L/D* = 0.23 configuration at Mach 5. In this chapter, the results from these computational studies will be discussed.

5.1 Steady-flowfield results

In this section computational results of the two problems will be represented in comparison with the flow visualization of the experiments. Besides pressure, Mach and temperature contours will be represented.

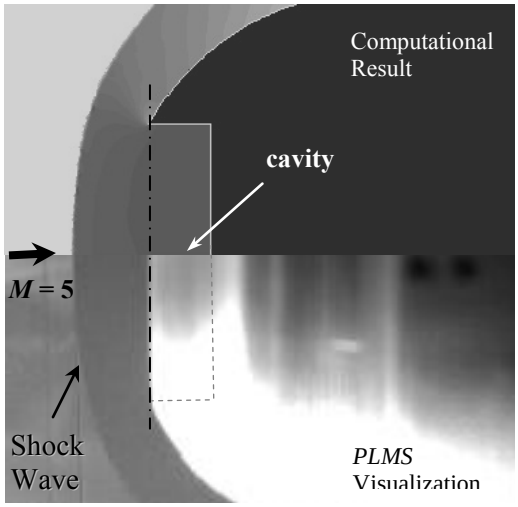


Fig. 5.1: PLMS Visualization [7] and Computational Result of *L/D* = 0.23, *M* = 5

The *PLMS* (Planar Laser Mie Scattering**)** visualization result of *L/D* = 0.23, *M* = 5 experiment [7] and static Pressure contours carried out by the computational analysis were represented in Figure 5.1. The computational result shows very good agreement with the experimental study in terms of shock positions and shapes.

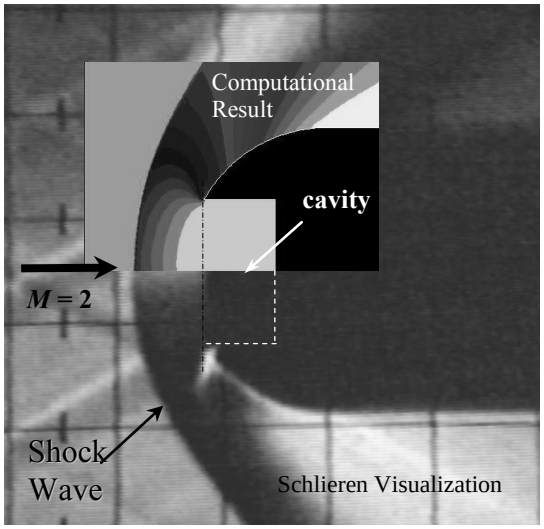


Fig. 5.2: Schlieren Visualization [8] and Computational Result of *L/D* = 0.5, *M* = 2

Schlieren visualization result of *L/D* = 0.5, *M* = 2 experiment [8] and static pressure contours carried out by the computational analysis were represented in Figure 5.2. The computational result shows very good agreement with the experimental study in terms of shock positions and shapes. Note that shock wave at Mach 5 is more straight than the shock wave at Mach 2.

Assuming inviscid flow, stagnation pressures behind the shockwaves can be calculated as 140.4 kPa for *L/D* = 0.23 configuration and 180.2 kPa for *L/D* = 0.5 configuration using Normal Shock Relations. In this computational analysis, they were computed as 139.3 kPa for *L/D* = 0.23 configuration and 178.5 kPa for *L/D* = 0.5 configuration. Considering viscous losses, these results are meaningful physically. Fig. 5.3 and Fig 5.4 contains the static pressure contours of the configurations.

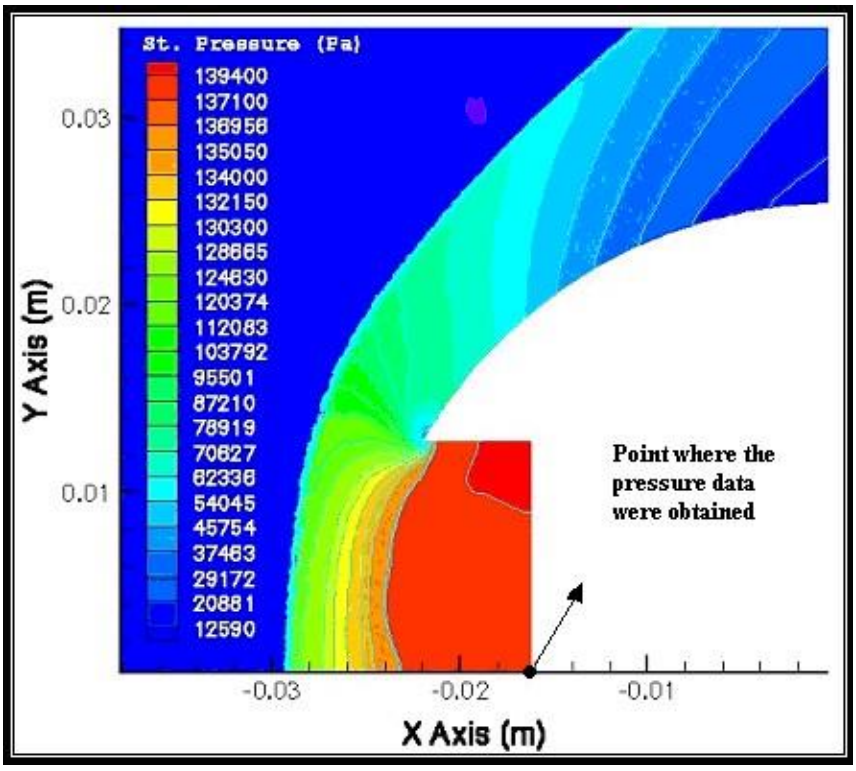


Fig 5.3: Static Pressure Contours of *L/D* = 0.23, *M* = 5

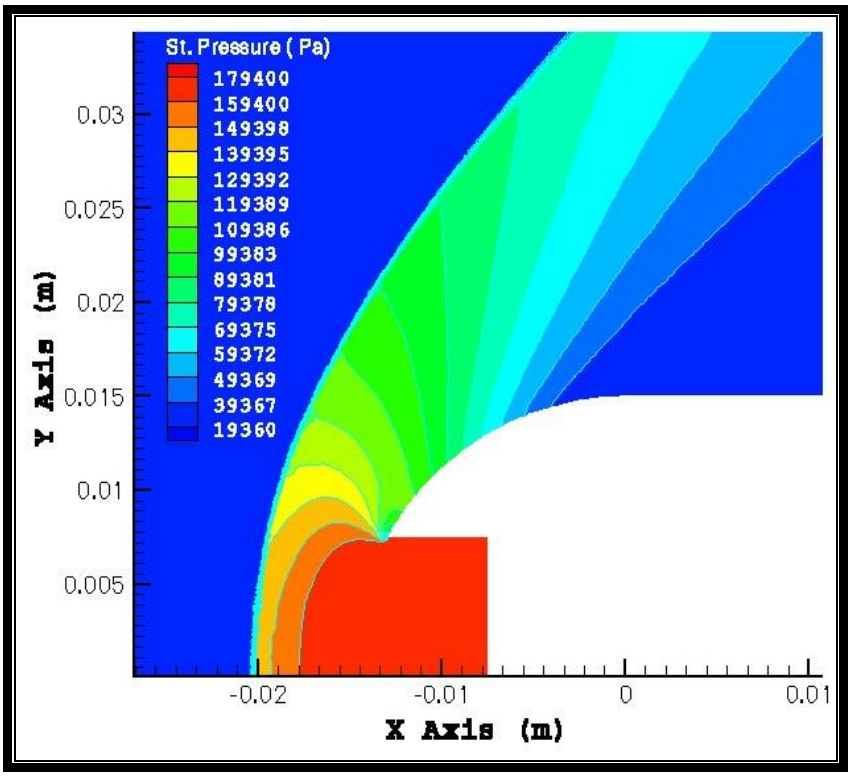


Fig 5.4: Static Pressure Contours of *L/D* = 0.5, *M* = 2

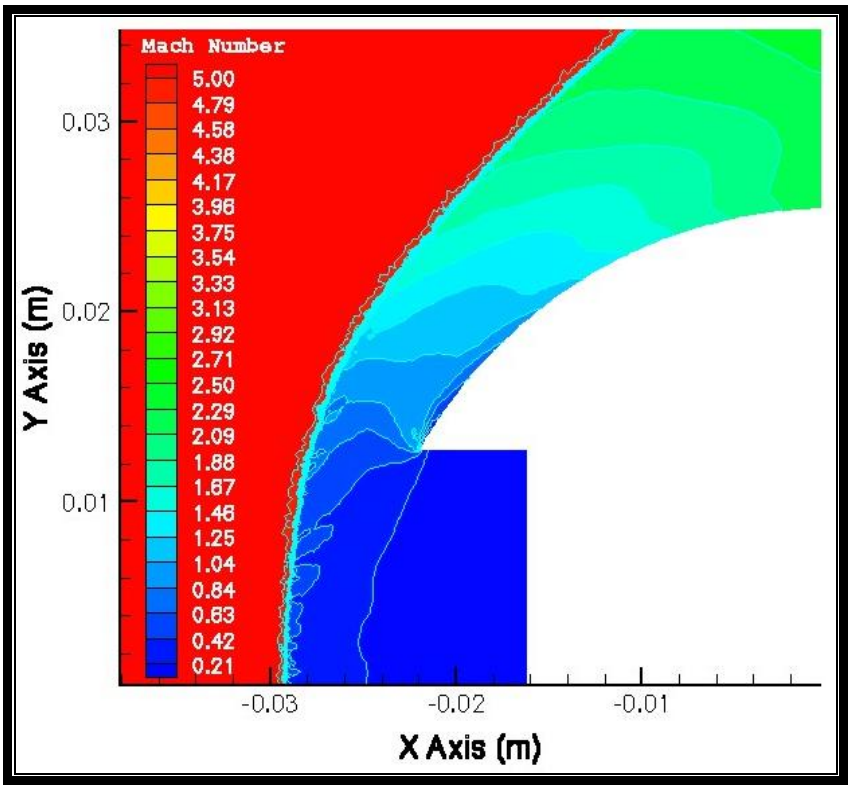


Fig.5.5: Mach Contours of L/D = 0.23, M = 5

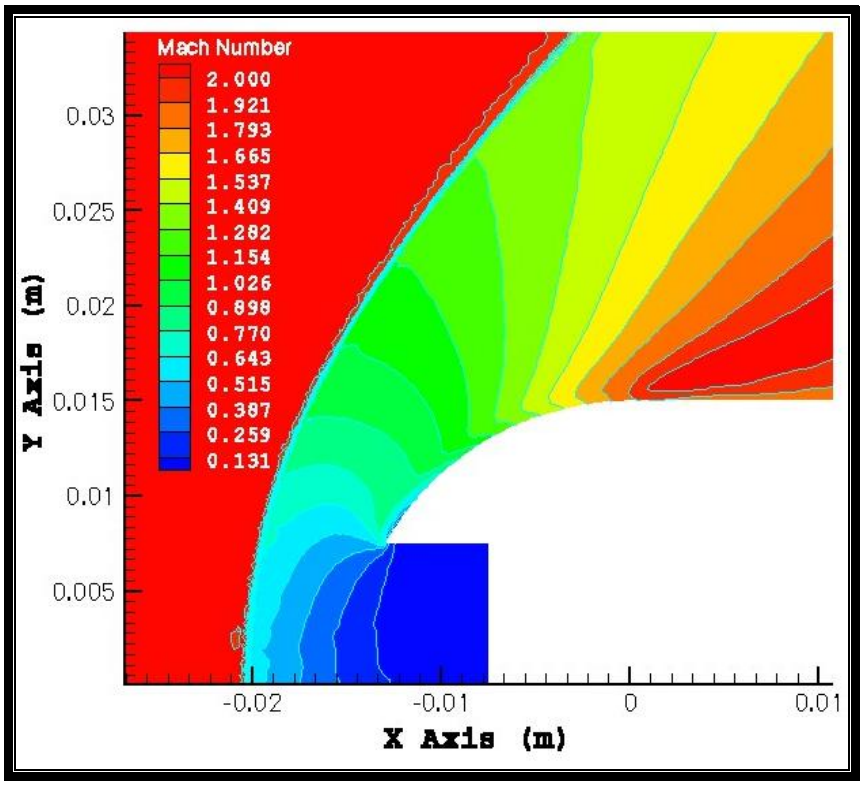


Fig 5.6: Mach Contours of L/D = 0.5, M =2

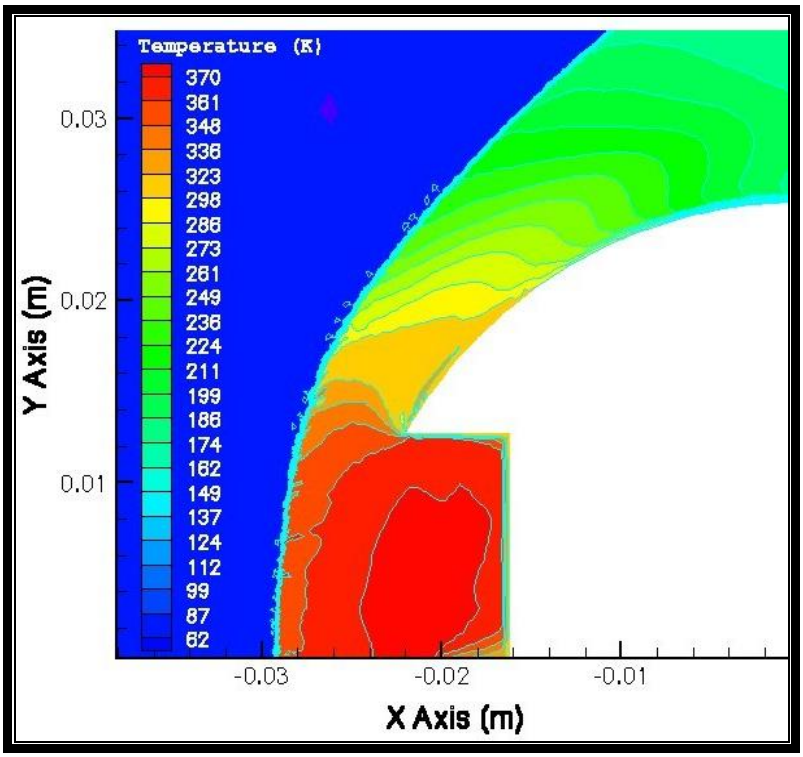


Fig 5.7: Temperature Contours of *L/D* = 0.23, *M* = 5

Mach contours are displayed in Fig. 5.5 and Fig. 5.6; Temperature contours are displayed in Fig. 5.7 and Fig. 5.9. Velocity vectors colored by temperature for *L/D* = 0.23, *M* = 5 were displayed in Figure 5.8. Note that the temperature in the vicinity of the cavity lip is relatively low.

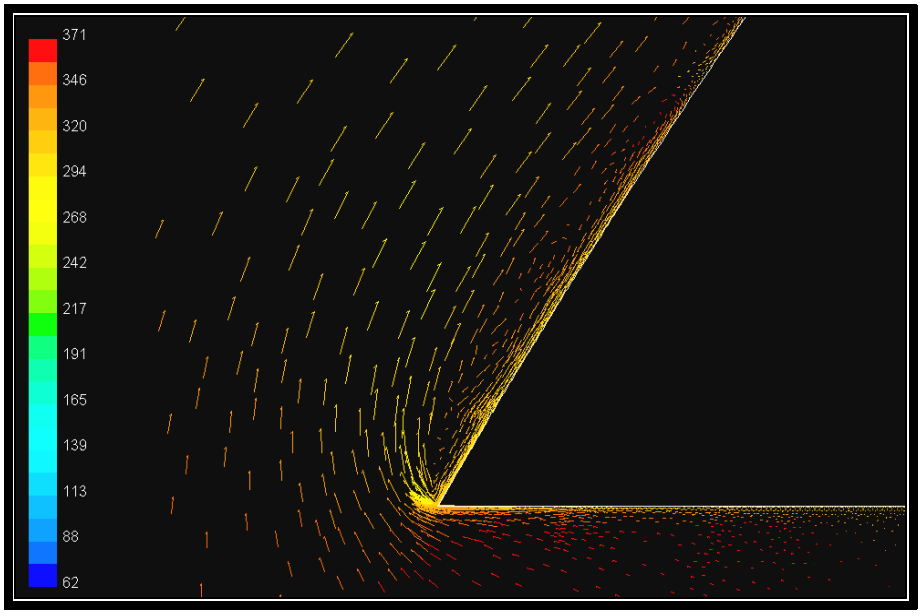


Fig 5.8:Velocity vectors colored by temperature for *L/D* = 0.23, *M* =5

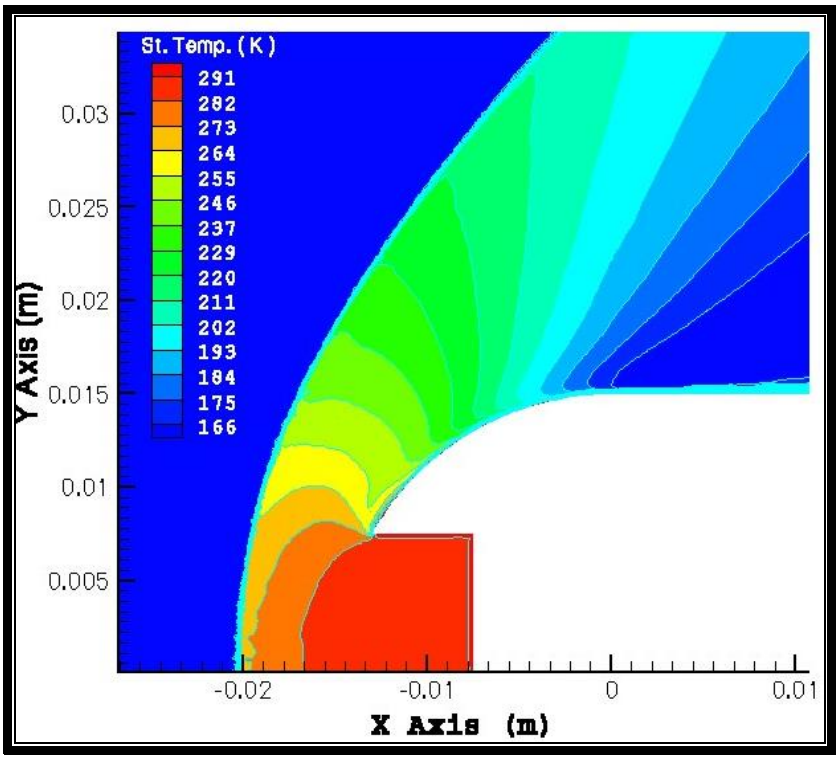


Fig 5.9: Temperature contours of L/D = 0.5, M = 2

5.2. Time-accurate simulation results

A sinusoidal freestream inflow pressure perturbation, with a peak to peak amplitude of H0.02 $P_э$ was simulated to understand the effect of freestream noise (e.g., unsteady freestream pressure at the inflow boundary) to the resonance mechanism for L/D = 0.23 shallow configuration. 4300 Pa was set for the pressure-far-field boundary as it was mentioned in Table 4.1.

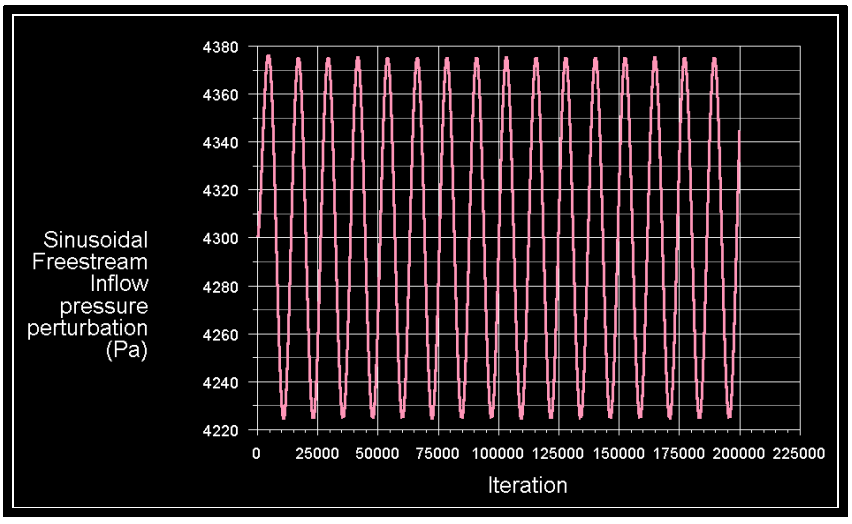


Fig. 5.10: Sinusoidal freestream inflow pressure oscillation

In these time-accurate simulations it was oscillated between 4222 Pa and 4378 Pa. A point was chosen in the pressure-far-field boundary to measure this inflow pressure. The aim was to control the UDF effect to the solution procedure in FLUENT solver. Fig.5.10 shows this pressure oscillation.

The primary mode was estimated at 7000 Hz from Eq. (4.6). Beginning from this frequency, 6250, 5800 and 20000 Hz. frequencies were tried in order to determine the distinctive resonant frequency. Inlet sinusoidal pressure frequencies, time steps and mesh cell numbers are displayed in Table 5.1.

Table 5.1: Information about the time-accurate simulations.

INLET SIN. PRESS. FREQUENCY	ITERATION #		TIME	MESH CELL NUMBER
7000 Hz	200000	1.54^{-04} sec.	30.8 sec.	40765
5800 Hz	200000	1.84^{-04} sec.	36.8 sec.	40765
6250 Hz	200000	1.00^{-08} sec	2.00^{-03} sec	82172
20000 Hz	200000	1.20^{-08} sec	2.40^{-03} sec.	82172

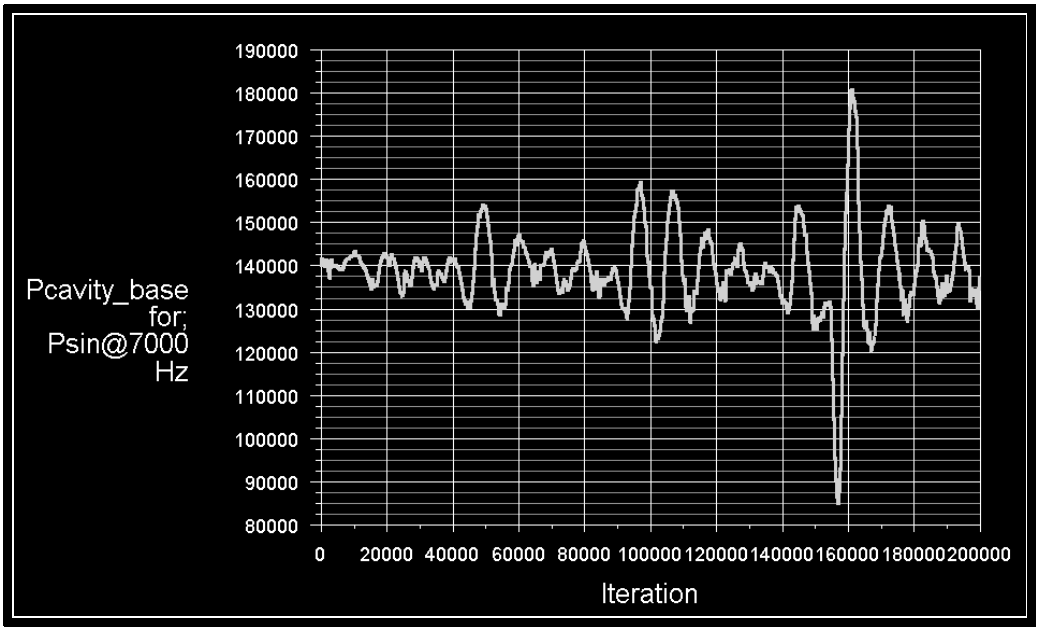


Fig 5.11: Stagnation pressure data for inflow pressure oscillation at 7000 Hz.

Although the primary mode frequency was estimated as 7000 Hz., the stagnation pressure oscillations at the base of the cavity did not result in periodic and large amplitude values at constant frequency. Therefore it is not the distinctive resonant frequency. Fig 5.11 shows these stagnation pressure data.

Fig. 5.12 contains power spectrum in normalized units for *L/D* = 0.23 (experimental). Var is the square of the standard deviation. Referring the experimental study, 6250 Hz was tried and nearly sinusoidal, periodic oscillations were observed at this frequency.

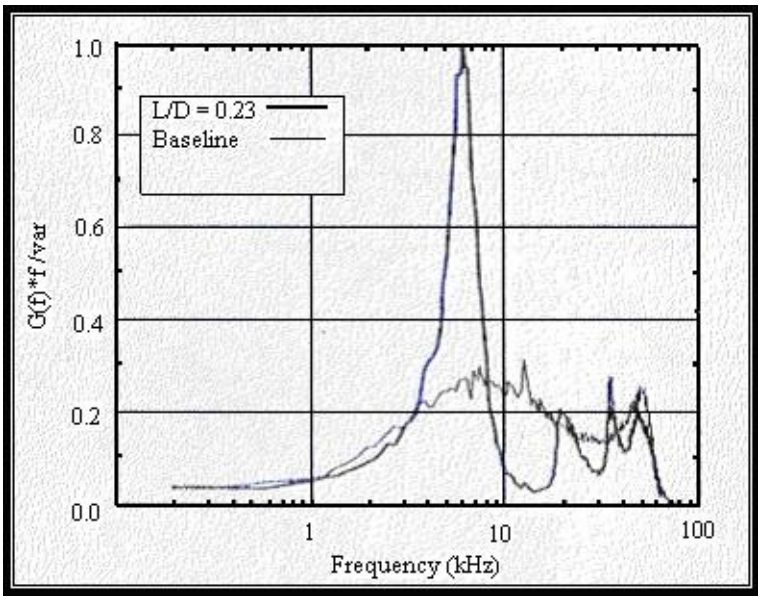


Fig 5.12: Power spectrum for *L/D* = 0.23 (experimental) [6]

To obtain more accurate result, mesh was refined and another run was performed. It took approximately for 111 hours (Total Wall Clock Time) with two 400 Mhz processors. Fig. 5.13. contains the result of this run.

The amplification is the ratio of the output (centreline base pressure) amplitude to the input (freestream) amplitude:

$$G = \frac{\left| p_{base\cdot\max} - \overline{p}_{base} \right|}{\left| P_{\infty\max} - \overline{p}_{\infty} \right|} \qquad (5.1)$$

A cavity of L/D 0.23, driven with a perturbation input of H0.02 P_{∞} at 6250 Hz, produced a perturbation output of approximately H0.04 P_{base} at 6250 Hz at the cavity base, for an amplification of 2. Hence, we can easily conclude that freestream noise in a small bandwidth of frequencies near the primary mode is the mechanism that drives resonant pressure oscillations within shallow forward-facing cavities.

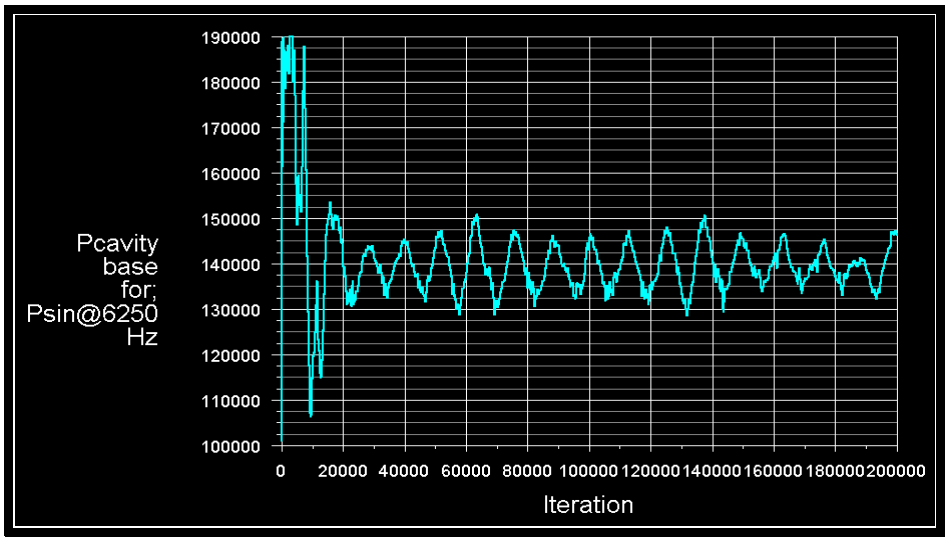


Fig 5.13: Stagnation pressure data for inflow pressure oscillation at 6250 Hz.

Two more runs at 5800 Hz and 20000 Hz were performed to verify the simulation. Figure 5.14 contains stagnation pressure data for sinusoidal inflow pressure oscillation at 5800 Hz and 6250 Hz. Both runs did not result in periodic and large stagnation pressure values at constant frequency at the base of the cavity.

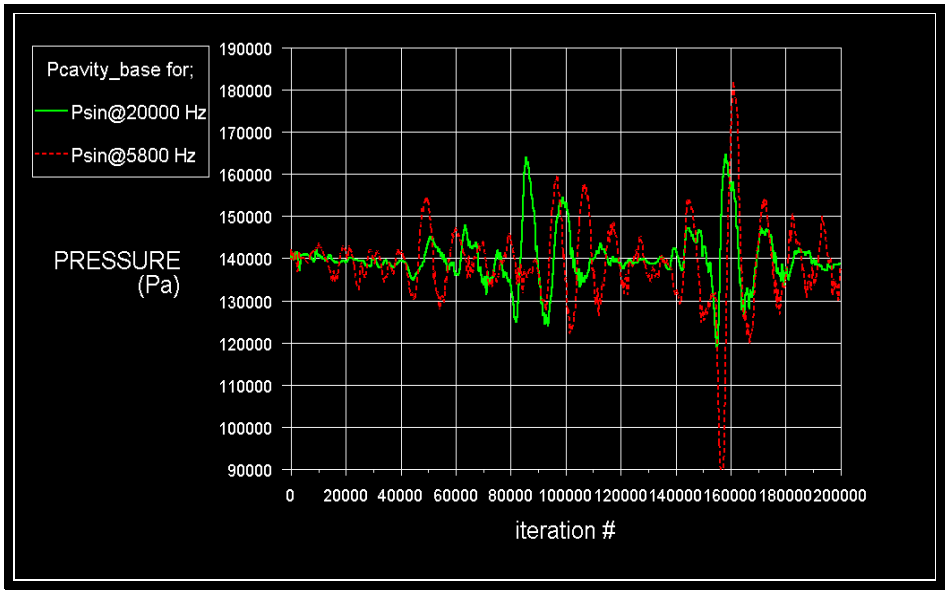


Fig 5.14: Stagnation pressure data for inflow pressure oscillation at 5800 Hz and 20000 Hz.

6. CONCLUSIONS

The first objective of this study was to obtain the steady-flowfield results for a forward-facing nose cavity on a blunt-nosed projectile both at Mach 5 and at Mach 2. The second objective was to obtain the time-accurate simulations of pressure oscillations for *L/D* = 0.23 configuration at Mach 5.The FLUENT solver, a finite volume code, was used in this computational study in order to achieve these two objectives. All of these computations were performed using High-Performance Computing Systems of ITU.

Static pressure contours obtained by the computational analysis were compared with the experimental flow visualization results qualitatively. The computational results showed very good agreement with the experimental flow visualization results in terms of shock positions and shapes.

Assuming inviscid flow, stagnation pressures behind the shockwaves can be calculated as 140.4 kPa for *L/D* = 0.23 configuration and 180.2 kPa for *L/D* = 0.5 configuration using Normal Shock Relations. In this computational analysis, they were computed as 139.3 kPa for *L/D* = 0.23 configuration and 178.5 kPa for *L/D* = 0.5 configuration. Considering viscous losses, these results are meaningful physically.

In the experimental studies performed by Yüceil and Dolling, resonant pressure oscillations have been observed in a forward-facing cavity in supersonic flow. In time-accurate simulations of pressure oscillations, a sinusoidal oscillation was implemented at inflow boundary to simulate freestream noise. A cavity of *L/D* 0.23, driven with a perturbation input of H0.02 $P_э$ at 6250 Hz, produced a perturbation output of approximately H0.04 P_{base} at 6250 Hz at the cavity base, for an amplification of 2.

Hence, it is concluded that freestream noise in a small bandwidth of frequencies near the primary mode is the mechanism that drives resonant pressure oscillations within shallow forward-facing cavities.

In order to observe the effect of the cavity more precisely, future work on this subject should focus on the simulations with different cavity depths and lip geometries. Sharp lips may produce a recirculation region, which cools the outer surface. In addition to this, strong oscillations associated with deep cavities may produce an additional cooling effect.

REFERENCES

[1] **Sun Tzu,** 1983, The Art of War, front flap, Delacorte Press, USA

[2] http://www.washingtonpost.com/ac2/wp-dyn/A42716-2002Feb7?language=printer

[3] http://www.armada.ch/e/6-97/page6-97_34.htm

[4] **Anderson, J. D.,** 1991, Fundamentals of Aerodynamics, pp. 6-8, McGraw-Hill Inc., Singapore

[5] **Stallings, R. L., and Burbank, P. B.**, 1959, Heat Transfer and Pressure Measurements on a Concave-Nose Cylinder for a Mach Number Range of 2.49 to 4.44, *NASA TMX-221*

[6] **Yüceil, K. B., and Dolling, D. S**., 1995, Nose Cavity Effects on Blunt Body Pressure and Temperature at Mach 5, *Journal of Thermophysics and Heat Transfer*, **9**, 612-619

[7] **Yuceil, K. B., and Dolling, D. S**., 1997, Infrared Imaging and Shock Wave Visualization over Concave Bodies, *Journal of Thermophysics and Heat Transfer*, **11**, 375-383

[8] **Fenercioğlu, İdil,** 2002, An experimental investigation of aerodynamic characteristics of a high speed projectile with a forward-facing nose cavity, *M. Sc. Thesis,* Institute of Science and Technology, Istanbul

[9] **Huebner**, **L.D. and Utreja, L.R.,** 1987, Experimental Flowfield Measurements of a Nose Cavity Configuration, *Society of Automotive Engineers,* 871880

[10] **Sambamurthi, J. K., Huebner, L.D., and Utreja, L.R**., 1987, Hypersonic Flow Over a Cone with Nose Cavity, *AIAA Paper*, 87-1193

[11] **Marquart, E. J., Grubb, J. B., and Utreja, L. R.**, 1987, Bow Shock Dynamics of a Forward-Facing Nose Cavity, *AIAA Paper,* 87-2709

[12] **Huebner, L.D., and Utreja, L.R.**, 1993, Mach 10 Bow-Shock Behaviour of a Forward-Facing Nose Cavity, *Journal of Spacecraft and Rockets*, **30**, 291-297

[13] **Bastianon, R. A**., 1968, Unsteady Solution of the Flowfield over Concave Bodies, *AIAA Paper,* 68-946

[14] **Bohachevsky, I. O. and Kostoff, R. N.,** 1972, Supersonic flow over Convex and Concave Shapes with Radiation and Ablation Affects, *AIAA Paper, 72-1024*

[15] **Yang, H.Q. and Antonison M.**, 1995, Unsteady Flowfield over a Forward-Looking Endoatmospheric Hit-to-Kill Interceptor, *Journal of Spacecraft and Rockets*, **32**, 440-444

[16] **Engblom, W. A., Yüceil, K. B., Goldstein, D. B., and Dolling, D. S**., 1996, Experimental and Numerical Study of Hypersonic Forward-Facing Cavity Flow, *Journal of Spacecraft and Rockets*, **33**, 353-359

[17] **Hoffmann K. A. and Chiang S. T.,** 1993, Computational Fluid dynamics for Engineers Volume II, pp. 265-267, Publication of Eng. Education Sys, Kansas

[18] **Anderson, J. D.,** 1982, Modern Compressible Flow, pp. 52, McGraw-Hill, Singapore

[19] **Anderson, J. D.,** 1989, Correlations for Hypersonic Shock Wave Shapes, *Hypersonic and High Temperature Gas Dynamics,* McGraw-Hill, Singapore

[20] **Jameson A., Schmidt W., Turkel E.,** 1981, Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes, *AIAA 14th Fluid and Plasma Dynamics Conference,* Palo Alto, California

[21] **Stetter, H.J**., 1973, Analysis of Discretization Methods for Ordinary Differential Equations, pp.176, Springer-Verlag

[22] http://www.darwinmag.com/learn/curve/column.html?ArticleID=45

APPENDICES

APPENDIX A : Creating The Geometry and Mesh Generation in Gambit Software

APPENDIX B : Parallel Processing in Fluent Software

APPENDIX C : User Defined Functions in Fluent Software

***All the information related to GAMBIT and FLUENT are taken from FLUENT MANUAL 5.5**

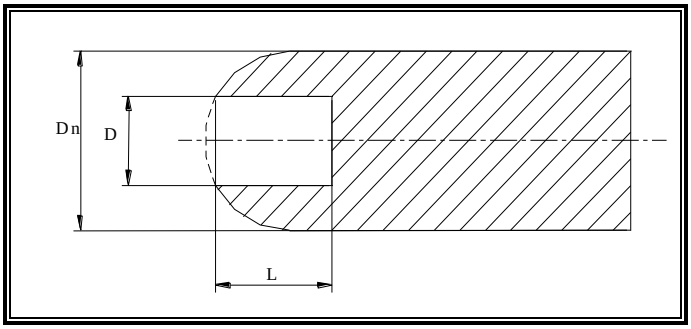
APPENDIX-A

CREATING THE GEOMETRY AND MESH GENERATION IN GAMBIT SOFTWARE

1. Overview
2. Creating The Geometry
3. Mesh Generation

APPENDIX-A

1. Overview



The model geometries studied are hemispherically blunted-cylinders with axisymmetric streamwise circular nose cavities.Two different cavity depths (L/D=0.5 in Mach 2 and L/D=0.23 in Mach 5) are modeled in these geometries.

L/D	0.5	0.23
D_n [mm]	30	50.8
D [mm]	15	25.4
L [mm]	7.5	5.842

. In this appendix, a 2-D mesh with L/D = 0.5 cavity depth will be built using a "bottom-up" approach. Since the other configuration with L/D = 0.23 has the same procedure , only the coordinates of its vertices will be mentioned. The "bottom-up" approach means that you will first create some vertices, connect the vertices to create edges, and connect the edges to make faces (in 3-D, you would stitch the faces together to create volumes). While this process by its very nature requires more steps, the result is a valid geometry

that can be used to generate the mesh. GAMBIT Modeling Guide is used as the main reference book throughout this procedure.

2. Creating The Geometry

Step 1: Select a Solver

1. Choose the solver you will use to run your CFD calculation by selecting the following from the main menu bar:

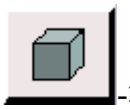
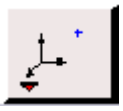
Solver ->FLUENT 5

This selects the FLUENT 5 solver as the one to be used for the CFD calculation. The solver currently selected is indicated at the top of the GAMBIT GUI (Gambit Graphical User Interface).

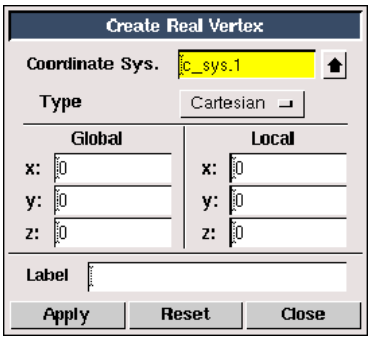
Step 2: Create the Vertices

1. Create vertices to define the outline of the model geometry.

a) Click the following command buttons.

OPERATION  -> GEOMETRY  -> VERTEX 

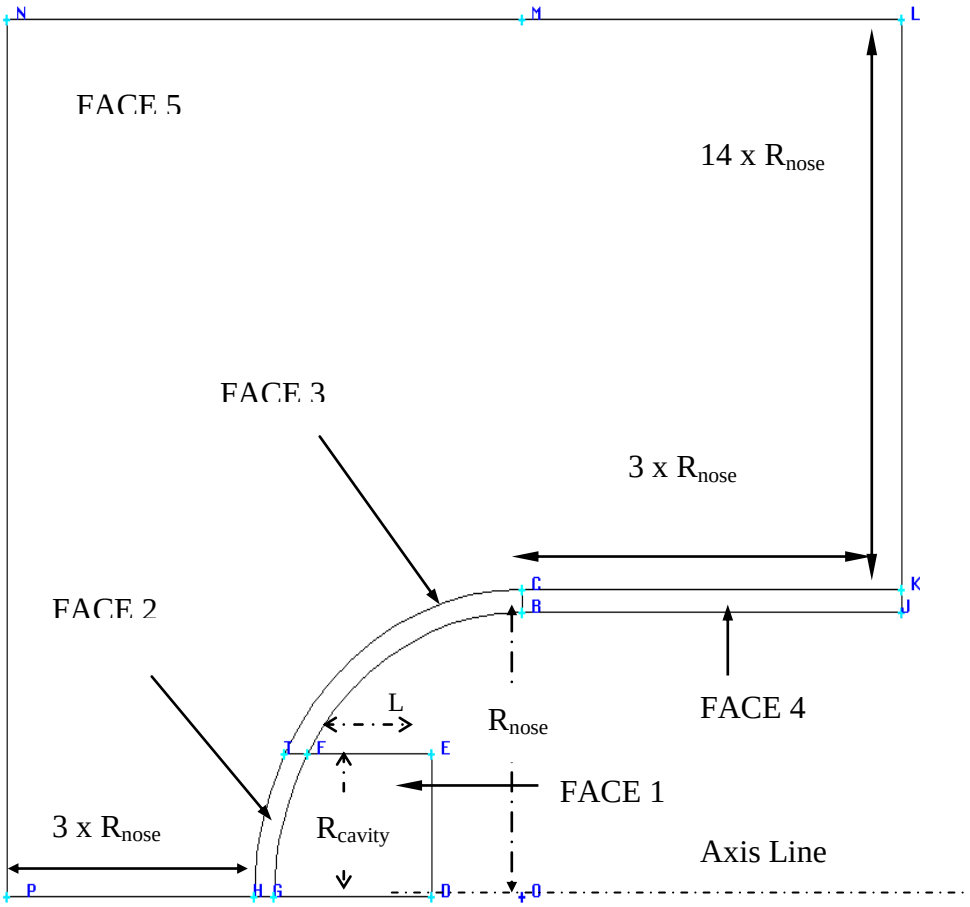
b) Fill in the following Create Real Vertex Form to specify the position of vertices. Use the Cartesian Coordinate system.



c) Click Apply to accept the selected coordinates and create vertices. The Vertices positions of both model geometries are listed below.You can use the following explanatory figure as well.

APPENDIX-A

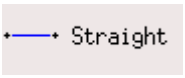
POSITIONS IN CARTESIAN SYSTEM (Meter)					
VERTICES FOR L/D = 0.5			VERTICES FOR L/D = 0.23		
Vertex	X Axis	Y Axis	Vertex	X Axis	Y Axis
O	0	0	O	0	0
B	0	0,015	B	0	0,0254
C	0	0,0162	C	0	0,027432
D	(-) 0,0054904	0	D	(-) 0,016155	0
E	(-) 0,0054904	0,0075	E	(-) 0,016155	0,0127
F	(-) 0,0129904	0,0075	F	(-) 0,021997	0,0127
G	(-) 0,015	0	G	(-) 0,0254	0
H	(-) 0,0162	0	H	(-) 0,027432	0
T	(-) 0,0143593	0,0075	T	(-) 0,0243151	0,0127
J	0,045	0,015	J	0,0762	0,0254
K	0,045	0,0162	K	0,0762	0,027432
L	0,045	0,2262	L	0,0762	0,383032
M	0	0,2262	M	0	0,383032
N	(-) 0,0612	0,2262	N	(-) 0,103632	0,383032
P	(-) 0,0612	0	P	(-) 0,103632	0



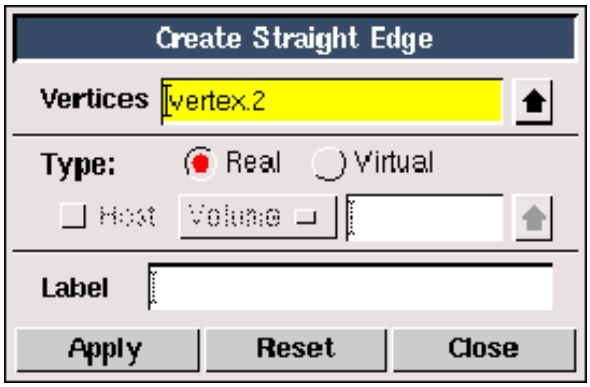
Step 3: Create the Edges

1. Create straight edges for the model geometry..

a) Click the following command buttons.

GEOMETRY  -> EDGE  -> CREATE EDGE 

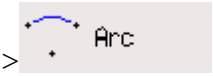
b) Fill in the following Create Straight Edge form to create straight edges connecting vertices. You can *Shift*-left-click to mark the vertices as well.



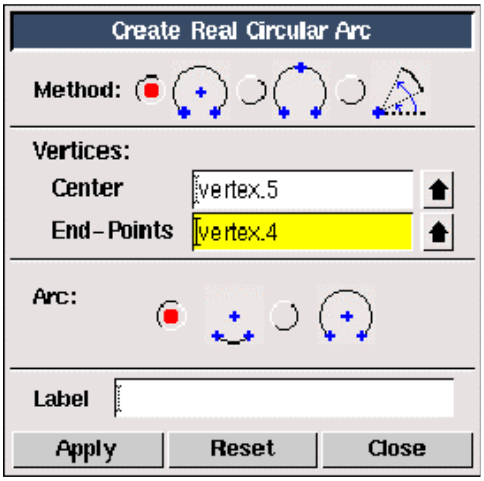
c) Click Apply to accept the selected vertices and create straight edges.

2. Create Arcs for the Nose of the Model Geometry.

a) Click the following command buttons.

GEOMETRY  ->EDGE  -> CREATE EDGE (Right-Click)  -> 

b) Fill in the following Create Real Circular Edge form to create arcs for the nose of the model geometry. Use Vertex 0 as center and Vertices B, G and C, H as end-points.

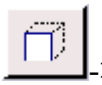


b) Click Apply to accept the selected vertices and create circular arcs.

Step 4: Create Faces from Edges

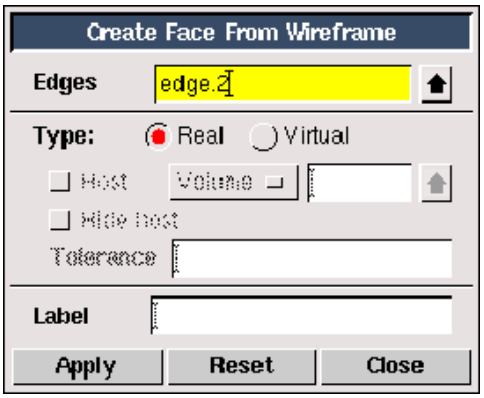
1.Form faces from existing edges.

a) Click the following command buttons.

GEOMETRY  -> FACE  -> FORM FACE 

b) *Shift*-left-click each edge of the model geometry, in turn, to form a continuous loop.

If you select an incorrect edge, click Reset in the Create Face From Wireframe form to deselect all edges, and then reselect the correct edges. Note that the edges must form a continuous loop, but they can be selected in any order. An alternative method to select several edges is to *Shift*-left-drag a box around the edges. The box does not have to completely enclose the edges; it only needs to enclose a portion of an edge to select it. The edges will be selected when you release the mouse button.



c) Click Apply to accept the selected edges and create faces.

2. Mesh Generation.

Step 5: Specify the Node Distribution

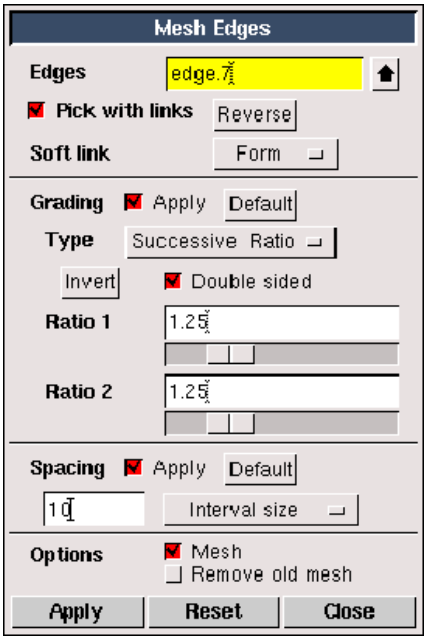
The next step is to define the grid density on the edges of the geometry. You will accomplish this graphically by selecting an edge, assigning the number of nodes, and specifying the distribution of nodes along the edge. An example of meshing the edge will be explained first. Then the other edge values needed for Mesh Edges form will be given in a table.

1. Specify the node density on the edges.

a) Click the following command buttons.

MESH  -> EDGE  -> MESH EDGES 

b) *Shift*-left-click the edge marked EF in explanatory figure of the model geometry given in Step 2. The edge will change color and an arrow and several circles will appear on the edge.



c) Check that Apply is selected to the right of Grading in the Mesh Edges form and that Successive Ratio is selected in the Type option menu. The Successive Ratio option sets the ratio of distances between consecutive points on the edge equal to the specified Ratio.

d) Enter 1.0 in the text entry box to the right of Ratio. Alternatively, you can slide the Ratio slider box (the small, gray rectangle with a vertical line in its center that is located on the slider bar) until 1.0 is displayed in the Ratio text box.

e) Be Careful not to select the Double sided check box under Grading.If you specify a Double sided grading on an edge, the element intervals are graded in two directions from a starting point on the edge. GAMBIT determines the starting point such that the intervals on either side of the point are approximately the same length. Note that Ratio changes to Ratio 1 and Ratio 2 when you select the Double sided check box. In addition, the value you entered for Ratio is automatically entered into both the Ratio 1 and the Ratio 2 text entry boxes.

f) Select Interval count from the option menu under Spacing and enter a value of 20 in the text entry box. Check that Apply is selected to the right of Spacing. GAMBIT will create 20 intervals on the edge.

g) Click the Apply button at the bottom of the form.

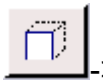
EDGE VALUES FOR MESHING					
EDGE	INTERVAL COUNT	RATIO	EDGE	INTERVAL COUNT	RATIO
BC	5	1	PH	12	0,74
CK	56	0,968	HG	5	1
KJ	5	1	GD	26	1
BJ	56	0,968	DE	20	1
KL	16	1,064	FG	20	1
MN	3	0,968	FB	56	1
NP	21	1	CT	56	1
TH	20	1	LM	2	0,8722
TF	5	1			

Step 6: Create Unstructured Meshes on Faces

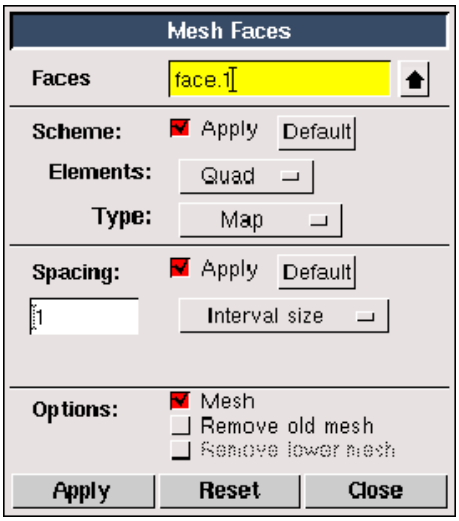
1. Create unstructured meshes that include triangular mesh elements in the faces.

APPENDIX-A

a) Click the following command buttons.

MESH  **-> FACE**  **-> MESH FACES** 

b) *Shift*-left-click the all node-marked faces.Check that **Apply** is selected to the right of **Scheme** in the **Mesh Faces** form and that Tri is selected in the **Elements** option menu. Then Type will automatically turn into Pave Type.**Tri** specifies that the mesh includes only triangular mesh elements and **Pave** warns that it creates an unstructured grid of mesh elements.



c) Click the **Apply** button at the bottom of the form.

Step 7: Set Boundary Types

1. Remove the mesh from the display before you set the boundary types. This makes it easier to see the edges and faces of the geometry. The mesh is not deleted, just removed from the graphics window.

a) Click the SPECIFY MODEL DISPLAY ATTRIBUTES command button  at the bottom of the Global Control toolpad.

b) Select Off from the option menu to the right of Mesh near the bottom of the form.

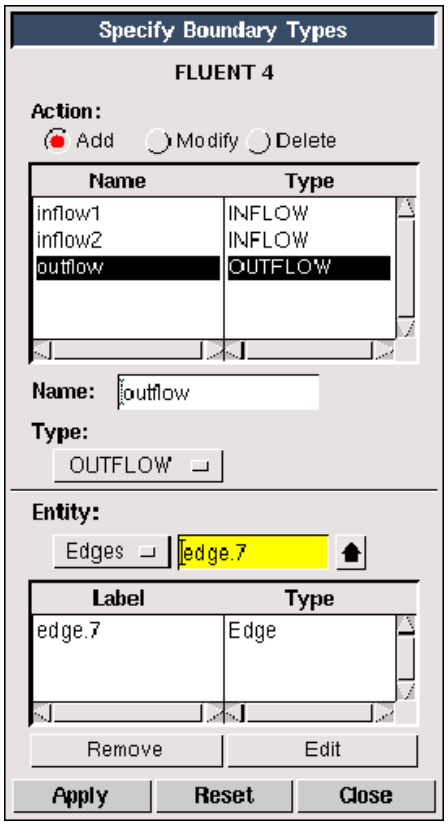
c) Click Apply and close the form.

2. Set boundary types for the model geometry.

APPENDIX-A

a) Click the following command buttons. Note that FLUENT 5 is shown as the chosen solver at the top of the form. The Specify Boundary Types form displays different Types depending on the solver selected.

ZONES  -> SPECIFY BOUNDARY TYPES 



b) Define axis boundary for the surface of the body.

i. Enter the name axis in the Name text entry box. If you do not specify a name, GAMBIT will give the boundary a default name based on what you select in the Type and Entity lists.

ii. Select AXIS in the Type option menu.

iii. Change the Entity to Edges by selecting Edges in the option menu below Entity.

iv. *Shift*-left-click the edges DG, GH, HP and click Apply to accept the selection. These edges will be set as an axis boundary.

c) Define axis boundary for the axis line.

i. Enter wall in the Name text entry box.

ii. Check that WALL is selected in the Type option menu and select the edges DE, EF, FB, BJ. Click Apply to accept the selection of the edges. These edges will be set as an wall boundary.

d) Define Pressure Far_Field boundary for freee stream conditions..

i. Enter Pressure Far_Field in the Name text entry box.

ii. Change the Type to PRESSURE_FAR_FIELD by selecting PRESSURE_FAR_FIELD in the option menu below Type.

iii. Select the edges PN, NM, ML and click Apply to accept the selection. These edges will be set as a Pressure Far_Field boundary.

e) Define Pressure_outlet boundary condition that permits flow to exit the solution domain.

i. Enter Pressure_outlet in the Name text entry box.

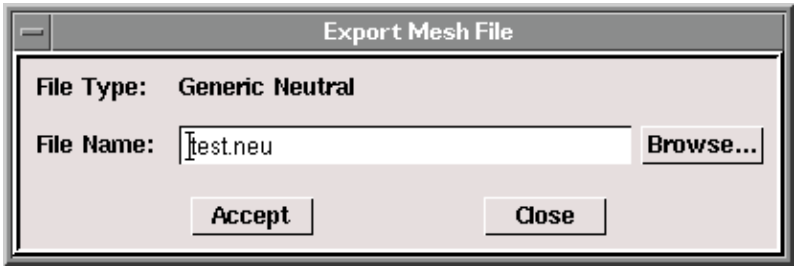
ii. Check that PRESSURE_OUTLET is selected in the Type option menu and select the edges JK and KL. Click Apply to accept the selection of the edges. These edges will be set as an Pressure_outlet boundary.

Step 8: Export the Mesh and Save the Session

1. Export a mesh file for the model geometry with forward-facing cavity.

File -> Export -> Mesh…

This opens the Export Mesh File form. *Note that the* File Type *is* UNS / RAMPANT / FLUENT 5.



a) Enter the File Name for the file to be exported.(for example, blunt.msh)

b) Click Accept. The file will be written to your working directory.

2. Save the GAMBIT session and exit GAMBIT.

APPENDIX-B

PARALLEL PROCESSING IN FLUENT SOFTWARE

1. Overview
2. Procedure for Using Parallel Fluent Solver

APPENDIX-B

1. Overview

When Ken Kennedy wants to explain to the layperson what parallel processing is all about, he points them toward the clothes hamper. If the laundry has been piling up, explains the director of the Center for Research on Parallel Computing in Houston, you can spend several hours washing it all in your machine at home. If you drive to the laundromat, however, and distribute it in several machines that all work at the same time, you shrink your time commitment by several hours. Those same principles are behind parallel processing, in which lots of processors-either in one computer or inside several linked machines-gang up to work on a single problem at one time. A typical desktop PC has one processor; a computer built to handle parallel processing can have several hundred. [22]

From this point of view, this computational study is performed in two different High-Performance computing systems of ITU. Very complicated, expensive but efficient and fast laundromats...

HARDWARE OF HIGH-PERFORMANCE COMPUTING SYSTEMS OF ITU	
BLUE	**ARTEMIS**
Silicon Graphics Origin 2000 Series,	Silicon Graphics Origin 3000 Series,
4 x 300 MHz R12000 processor,	8 x 400 MHz R12000 processor,
2 GB Memory 72 GB SCSI Disc	5 GB Memory 180 GB SCSI Disc

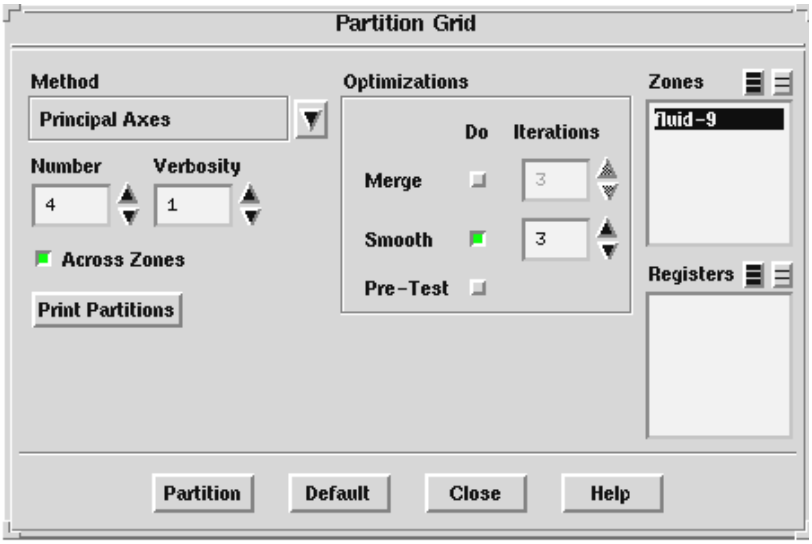
Irix 6.5.10 operating system and an optimized Message Passing Interface software is used in these Systems. FLUENT 5.5 Manual is used as the main reference book throughout this procedure.

2. Procedure for Using Parallel Fluent Solver

Step 1: Partition The Grid

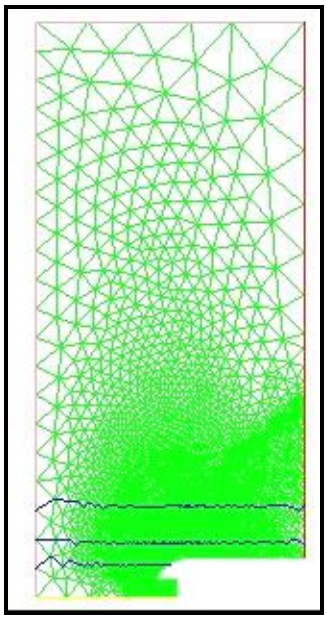
1. Using the menü optins, set the values for Partition Grid Panel.

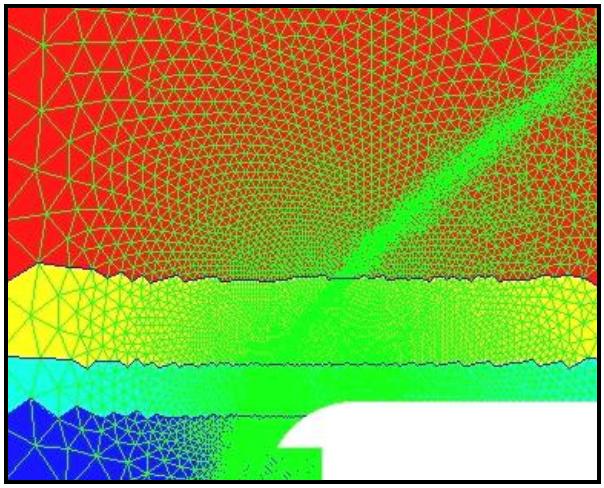
Grid -> Partition...



a) *Method* contains a drop-down list of the recursive bisection methods that can be used to create the grid partitions. The grid is partitioned using a bisection algorithm. The selected algorithm is applied to the parent domain, and then recursively applied to the child subdomains. For example, to divide the grid into four partitions, the solver will bisect the entire (parent) domain into two child domains, and then repeat the bisection for each of the child domains, yielding four partitions in total. To divide the grid into three partitions, the solver will "bisect" the parent domain to create two partitions---one approximately twice as large as the other---and then bisect the larger child domain again

to create three partitions in total. Cartesian Y-Coordinate was chosen as Grid Partitioning Method for this problem. It bisects the parent domain and all subsequent child subdomains perpendicular to the Y coordinate direction.





b) *Number* defines the desired number of grid partitions. This usually matches the number of processors available for parallel computing. Throughout this study 2 to 8 processors were used according to the availability in the workstations.

c) *Verbosity* specifies the amount of information to be reported in the text (console) window during the partitioning. With the default value of 1, the solver will print the number of partitions created, the number of bisections performed, the time required for the partitioning, and the minimum and maximum cell, face, interface, and face-ratio variations. If you increase the Verbosity to 2, the partition method used, the partition ID, number of cells, faces, and interfaces, and the ratio of interfaces to faces for each partition will also be printed in the console window. If you decrease the Verbosity to 0, only the number of partitions created and the time required for the partitioning will be reported. Since I have had the computer write the data in log files, generally 0 was chosen for verbosity to avoid unnecessary information.

d) *Across Zones* allows partitions to cross zone boundaries (the default). If turned off, it will restrict partitioning to within each cell zone. This is recommended only when cells in different zones require significantly different amounts of computation during the solution phase, for example if the domain contains both solid and fluid zones. Since all cells of the studied grid were in fluid zone, it wasn't turned off throught the computations.

e) *Print Partitions* prints the partition ID, number of cells, faces, and interfaces, and the ratio of interfaces to faces for each partition in the console window. In addition, it prints the minimum and maximum cell, face, interface, and face-ratio variations.

f) *Optimizations* contains toggle buttons for activating schemes to optimize the partitions created by the selected bisection method. In addition, the optimization scheme will be applied until appropriate criteria are met, or the maximum number of iterations have been executed. If the Iterations counter is set to 0, the optimization scheme will be applied until completion, without limit on the maximum number of iterations.*Merge* attempts to decrease the number of interfaces by eliminating orphan cell clusters (an orphan cluster is a group of connected cells whose members each have at least one face coincident with an interface boundary). *Smooth* attempts to minimize the number of interfaces by sacrificing cells on the partition boundary to the neighboring partition to reduce the partition boundary surface area. Both merge and smooth iterations were used in this study.

g) *Pre-Test* instructs the solver to test all coordinate directions and choose the one which yields the fewest partition interfaces for the final bisection.

h) *Zones* contains a list of cell zones. Partitioning will be applied to cells in zones selected from this list. Fluid (default) was chosen in the study.

i) *Registers* contains a list of cell registers that have been created using the adaption tools. You can restrict partitioning to a group of cells by selecting a register containing the cells. *Registers* property was not used.

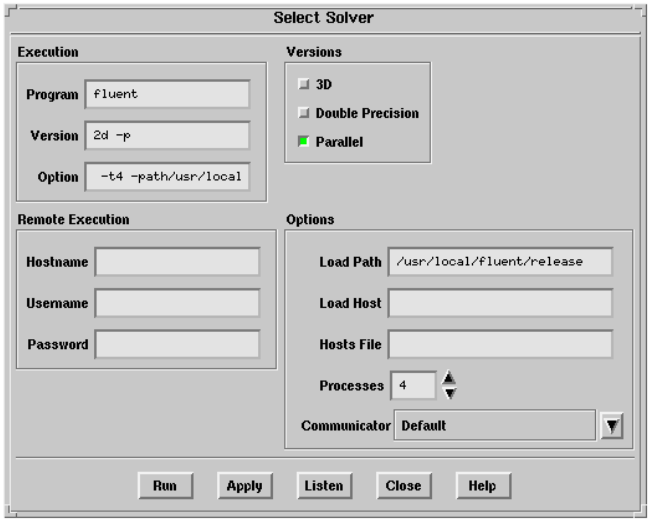
j) *Default* sets all controls to their default values, as assigned by FLUENT. After execution, the Default button becomes the Reset button. *Reset* resets the fields to their most recently saved values (i.e., the values before Default was selected).After execution, the Reset button becomes the Default button.

2) After you set all parameters, click the *Partition* command button that subdivides the grid into the selected number of partitions using the prescribed method and optimization(s).

Step 2: Start up the parallel solver

1) Type the usual startup command without a version (i.e., fluent), and then use the Select Solver Panel to specify the parallel architecture and version information.

File -> Run...



2. Under Versions, specify the 3D or 2D single- or double-precision version by turning the 3D and Double Precision options on or off, and turn on the Parallel option.

3. Under Options, select the message-passing library in the Communicator drop-down list. The Default library is recommended, because it selects the library that should provide the best overall parallel performance for your dedicated parallel machine.

4. Set the number of CPU's in the Processes field.

5. Click on the Run button to start the parallel version. No additional setup is required once the solver starts.

Step 3: Read in the partitioned case file, and specify any aspects of the problem definition that were not set in the serial solver.

Step 4: Calculate the solution.

! Note that for load balancing, partitioning in the serial solver should be repeated after any grid adaption.

APPENDIX-C

USER DEFINED-FUNCTIONS IN FLUENT SOFTWARE

1. Overview of User-Defined Functions
2. The UDF that is Used in the Time-Accurate Simulations of the Pressure Oscillations.
3. The Steps for Including a User-Defined Function in a FLUENT Calculation

APPENDIX-C

1. Overview of User-Defined Functions

User-defined functions can be used to enhance the standard features of FLUENT in a number of ways. In this study a Compiled UDF was written to oscillate the inlet pressure at given frequencies. So it was used to customize the boundary condition.

User-defined functions (UDFs) are written in the C programming language. There are two types of user-defined functions: interpreted and compiled. Interpreted UDFs are compiled at runtime from within a FLUENT session. Compiled UDFs are compiled and grouped in a shared library using a Makefile before you begin a FLUENT session. The shared library is then linked with the standard FLUENT executable at runtime. The standard FLUENT executable will remain unchanged, but you will be able to link one of any number of shared libraries to it to form effective custom executables.

One advantage of using compiled UDFs is that they run much faster than interpreted UDFs. Another advantage is that compiled UDFs are given complete access to the solver. Interpreted UDFs, on the other hand, have limited access to the solver and, for example, do not recognize C language structures. Interpreted UDFs are architecture-independent, and as a result, are more convenient to use.

To assist in the writing of user-defined functions, an extensive list of primitive functions is made available in FLUENT Tutorial. This list contains math functions and all problem variables, including, for example:

- velocities, density, temperature, and turbulence quantities
- x, y, and z coordinates of cell centers
- face areas and cell volumes

Once loaded, the user-defined functions can be selected from the regular user interface as boundary conditions, source terms, custom properties, etc. FLUENT 5.5 Manual is used as the main reference book throughout this procedure.

2. The UDF That Is Used In The Time-Accurate Simulations of The Pressure Oscillations.

Before you begin writing UDFs, the udf.h file needs to be accessible in your path. It contains the function declarations of all the functions available for your use. A large

assortment of math functions and problem variables is available. The location of this file is:

path/Fluent.Inc/fluent5.*x*/src/udf.h

where path is the directory in which you have placed the release directory, Fluent.Inc, and x is replaced by the appropriate number for the release you have (e.g., 5 for fluent5.5). In general, you should not copy udf.h from the release area. The compiler is designed to look for this file locally (in your current directory) first. If it is not found in your current directory, the compiler will look in the /src directory automatically. You should not, under any circumstances, alter the file udf.h.

UDF "sinusoidal_p_7000-86.c" Listing :

```
# include "udf.h"
# define pi 3.141592654
# define pavg 4300.0
# define amplitude 86.0
# define nampl 0.02
# define freqhz 7000
DEFINE_PROFILE(inlet_xp_sinus_b, thread, nv)
{
face_t  f;
real flow_time = RP_Get_Real("flow-time");
real freqrps = freqhz*2.0*pi;
real  press=pavg*(1.+nampl*sin(freqrps* flow_time)));
/*  printf ("p = %.5f  t = %.5e  \n" ,press, flow_time); */
begin_f_loop (f, thread)
{
  F_PROFILE (f, thread, nv) = press;
}
end_f_loop (f, thread)
}
```

APPENDIX-C

UDF sinusoidal_p_7000-86.c produces a perturbation input of ±0.02 $P_э$ at 7000 Hz in the time-accurate simulations of this study. Different frequencies were tried just by setting new values for the variable freqhz throughout the study. DEFINE_PROFILE states the inlet_xp_sinus_b boundary condition profile. Several data types are defined especially for use in FLUENT. These serve as pointers to objects and can be used when you are defining functions. Note that these definitions are case-sensitive. A Thread describes a boundary or cell zone. Examples include a fluid zone, a solid zone, or an inlet zone. A face_t corresponds to the face of a cell, and is the location where inlet boundary conditions would be defined, for example. The arguments of F_PROFILE are the indices of the face, a face's thread, and an integer, nvar. The integer is a numerical label for the variable that is being set at a particular boundary. For example, an inlet boundary may have a total pressure and a total temperature associated with it (both of which can be functions). One of these will be identified by the integer 0, and the other by the integer 1. If, instead, the inlet has three velocity components and a static temperature, they will be identified by integers 0 through 3, and so forth. The integer nvar is a quantity that you should not change. Its value will be passed to your UDF by FLUENT. begin_f_loop(f,thread) and end_f_loop(f,thread) state the loop over faces in a face thread. The macro F_PROFILE is used to set the value of a variable at a face. This function is used when you are generating boundary conditions. In addition to these, all quantities in a user-defined function must be specified in SI units and all user-defined function names must be specified in lower case (no capitals).

3. The Steps for Including a User-Defined Function in a FLUENT Calculation

Step 1 : Create a working directory. If you have a previously-created case file, copy it to your working directory.

Step 2 : Using a text editor, write the user-defined function “sinusoidal_p_7000-86.c” and save it in your working directory.

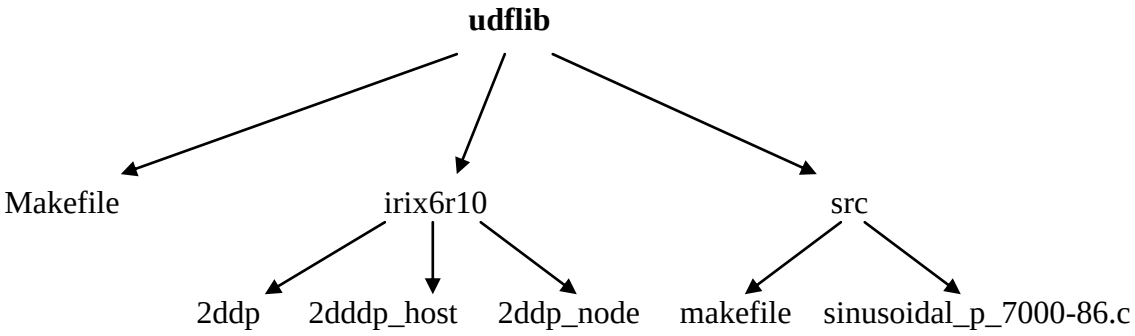
Step 3 : Build your compiled UDF library.

APPENDIX-C

1. There are two FLUENT files that are needed to build your compiled UDF library: Makefile and makefile. The file makefile has a user-modifiable section that allows you to enter your source functions, as well as the path to your FLUENT release directory (Fluent.Inc). Makefile is an executable that calls makefile. The complete path to each of these files is:

path/Fluent.Inc/fluent5.5/src/Makefile.udf

path/Fluent.Inc/fluent5.5/src/makefile.udf



a) Make a directory that will store your library (e.g., libudf).

b) Copy Makefile.udf from the directory shown above to your library directory (libudf), and name it Makefile.

c) Under the library directory you just created, make a source directory for your source code, and name it src.

d) Copy your source code (sinusoidal_p_7000-86.c) to your /src directory.

e) Copy makefile.udf from the directory shown above to your /src directory, and name it makefile.

f) Identify the architecture of the machine that you will run FLUENT on. To do this; Start FLUENT, Scroll up the FLUENT console window to the message that begins with " Starting" and identify the FLUENT architecture (irix6r10). Exit Fluent.

g) Make directories for the versions (2D or 3D) you want to build for your architecture.(irix6r10/ 2ddp, irix6r10/2dddp_host, irix6r10/2ddp_node)

h) Edit makefile to set the following parameters:

- SOURCES = the user-defined function(s) to be compiled
- FLUENT_INC = the path to your release directory

APPENDIX-C

An excerpt from a sample makefile is shown below:

```
#------------------------------------------------------------#
# makefile for user defined functions.
#
# sccs id:  @(#)makefile.udf     1.17 01/10/00
#------------------------------------------------------------#

#------------------------------------------------------------#
# User modifiable section.
#------------------------------------------------------------#
SOURCES= sinusoidal_p_7000-86.c
FLUENT_INC= /usr/local/Fluent.Inc

 #------------------------------------------------------------#
 # Build targets (do not modify below this line).
 #------------------------------------------------------------#
```

i) From your library directory (libudf), execute the Makefile by typing a command that begins with make and includes the architecture of the machine you will run FLUENT on that you identified in a previous step (irix6r10).

make "FLUENT_ARCH=irix6r10"

The following messages will be displayed:

```
For d in irix6r10/[23]*;do \
   ( \
     cd $d; \
     for f in ../../src/*.[ch] ../../src/makefile; do \
        if [ ! –f ‘basename $f’ ]; then \
           echo “# linking to” $f “in” $d; \
           ln –s $f .; \
        fi; \
      done; \
      echo “”; \
      echo “# building library in” $d; \
      make –k>makelog 2>&1; \
```

```
        cat makelog; \
      ) \
    done
```

building library in irix6r10/2ddp

make libudf.so “CFLAGS=-KPIC –wansi –fullwarn –0 –n32” “LDFLAGS=-shared-n32 –lm”

UX:make: INFO: ‘libudf.so’ is up to date.

building library in irix6r10/2ddp_host

make libudf.so “CFLAGS=-KPIC –wansi –fullwarn –0 –n32” “LDFLAGS=-shared-n32 –lm”

UX:make: INFO: ‘libudf.so’ is up to date.

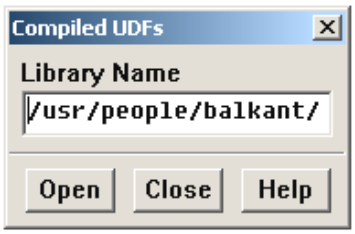
building library in irix6r10/2ddp_node

make libudf.so “CFLAGS=-KPIC –wansi –fullwarn –0 –n32” “LDFLAGS=-shared-n32 –lm”

UX:make: INFO: ‘libudf.so’ is up to date.

For the sample makefile above, the user-defined function sinusoidal_p_7000-86.c will be compiled and stored in the shared library named libudf.so for the versions you have specified. Although only one C function was used in this study, you can specify multiple sources under SOURCES = in the makefile.

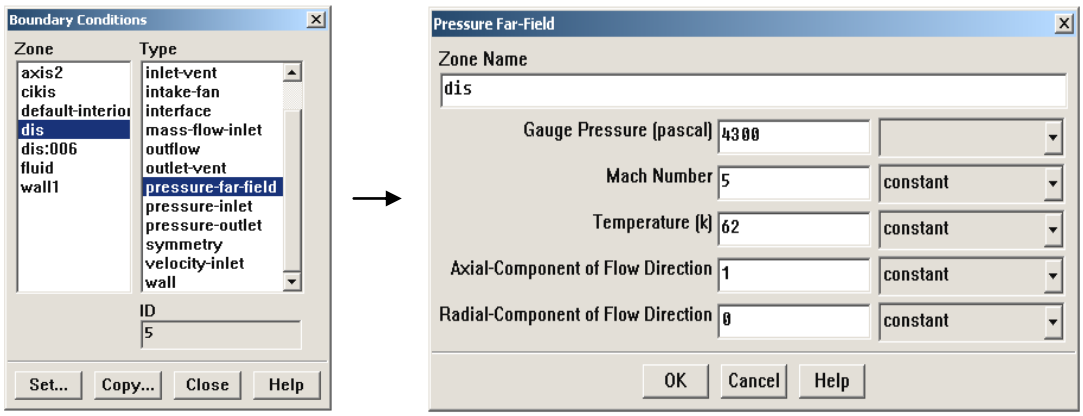
j) Start FLUENT and Read (or set up) your case file. In the Compiled UDFs Panel type the name of your library directory (udflib) or the complete path (/usr/people/balkant/udflib) under Library Name, and click Open to link it to the FLUENT executable.



APPENDIX-C

Define -> User-Defined -> Functions -> Compiled...

Step 4 : Once you have compiled the sinusoidal_p_7000-86.c, the function will become visible in the Pressure Far-Field Panel and can then be selected as Gauge Pressure [pascal] .



Define -> Boundary Coditions (*dis(zone)* + *pressure-far-field*) -> Pressure Far-Field

Step 5 : Write a case file after the UDF has been compiled and specified.The association with the shared udflib library will be saved with your case file.

CURRICULUM VITAE

Taner BALKAN was born in İzmir in April 1974. He completed his secondary schooling at the Maltepe Military High School in İzmir. He graduated from the Army Academy (ANKARA) in 1996 as an infantry officer. Upon commissioning, he underwent his officer's basic training at the infantry branch school in Eğridir (ISPARTA) and Tuzla (İSTANBUL) in 1997. He spent the next two years in Gökçeada, as Commando Platoon Commander position and the following year in Kars, as Border Platoon Commander position. In November 2000, he enrolled in the Defense Technologies Master of Science program of ITU.